GRID AND CLOUD COMPUTING FOR E-SCIENCE APPLICATIONS

POLITECNICO DI TORINO

SCUOLA DI DOTTORATO

Dottorato in Ingegneria Elettronica e delle Comunicazioni

Tesi di Dottorato

Grid and Cloud computing for E-science applications

Olivier Terzo

Tutore
Prof. Giovanni Perona

Coordinatore del corso di dottorato
prof. Ivo Montrosset

Marzo 2013

Ringraziamenti

A mia madre, Silvia e Myriam.

Indice

Elenco delle tabelle

Elenco delle figure

Capitolo 1

Infrastructures for eScience applications

1.1 e-Science applications

eScience fields which include areas such as spatial data, electromagnetic,bioinformatics, energy, social sciences, simulation, physical science have on the course of recent years a significant development regarding the complexity of algorithms and applications for data analysis. Information data has also evolved with an explosion in term of data volume and datasets for the scientific community. This has led researchers to identify new necessity regarding tools analysis, applications, by a profound change in computing infrastructures utilization.

The field of eScience is constantly evolving through the creation of ever more growing scientific community who have a real needs in availability in computational resources ever more powerful calculations. Another important issue is the ability to be able to share results, this is why cloud technology through virtualization can be an important help for the scientist community for giving a flexible and scalable IT infrastructure depending on necessities. Indeed, cloud computing allows for the provision of computing resources, storage in an easy configurable way and adaptable in functions of real needs. Researchers often do not have all the computing capacities to meet their needs, so cloud technology and cloud models as Private, Public and Hybrid is an enable technology for having a guarantee of service availability, scalability and flexibility.

The transition from traditional infrastructure to new virtualized with distributed models allows researchers to have access to an environment extremely flexible allowing an optimization of the use of hardware for having more available resources. However, the computational needs on e-Science have a direct effect regarding the

way that applications are developed. The approach of writing algorithm and applications is still too tied to a model centered on a workstation for example. The vast majority of researchers conducts the writing process of their applications on their laptop or workstation in a limited context of computing power, storage and in a non-distributed way.

Often after some first tests for consolidation a validation on larger data set is required and therefore needs a more suitable infrastructure and high availability of computing power resources. Therefore the elaboration time is very long and in some cases it is extremely difficult to proceed to validation using a big dataset due to the limited computing power and storage capacities own by developers, typically on notebook or workstations.

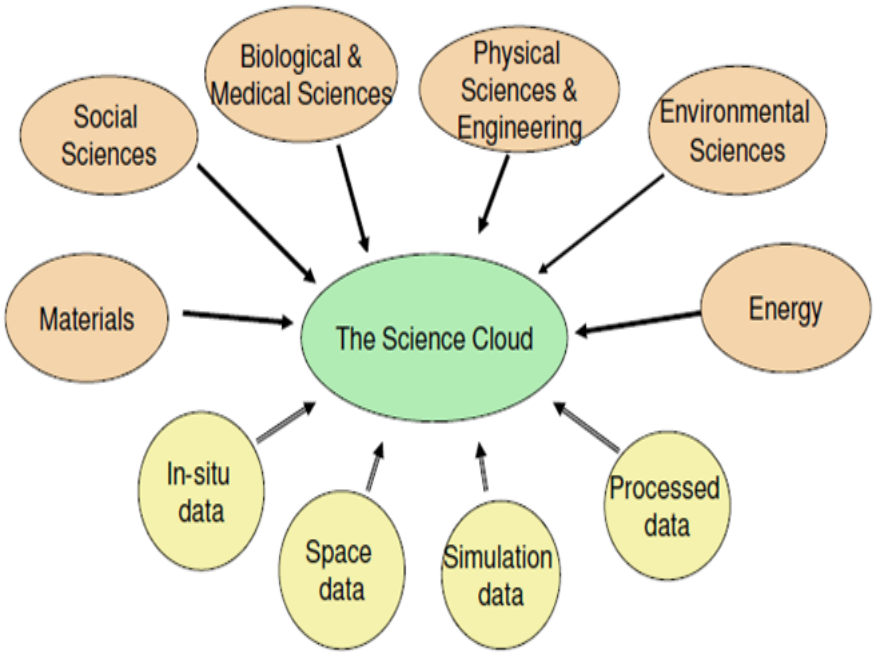

Figura 1.1. eScience Cloud

In fact it is necessary to proceed to a new approach for applications development directly integrable on modern computing infrastructure by adapting applications and algorithms in order to be scalable and easily exploitable on new IT technology such as cloud computing. We must rethink the way in which applications are developed because it provides a double benefit: first for code writing optimization for reducing process analysis time duration and more often it will be necessary to proceed by a

reverse engineering phase and the second benefit is on the optimization of the use of computational resources in a multiuser context.

1.2 Cloud Computing technology

1.2.1 Virtualization

Virtualization is a technology that allows running several concurrent Operating System (OS) instances inside a single physical machine called host. The physical hardware device is divided into multiple virtual environments called guest system. A Virtual Machine (VM) is an instance of the physical machine. On a single hardware more than one VM can be run it's a way to have more Operating systems running on the same hardware and the hypervisor, or Virtual Machine Manager (VMM), allows multiple operating systems to concurrently run on a single host computer. It is so named because it is conceptually one level higher than a supervisory program.

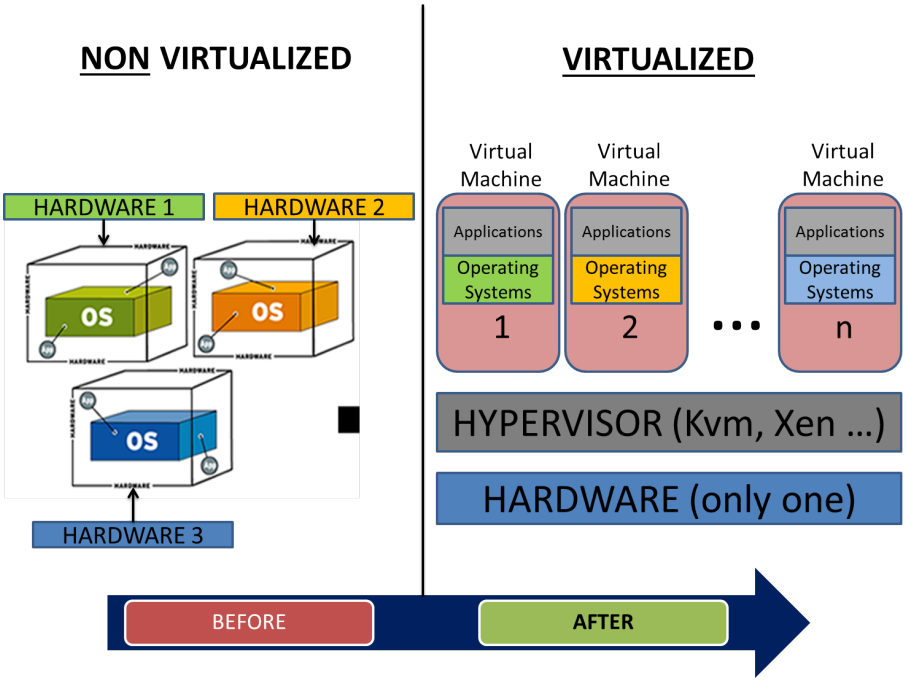

Figura 1.2. Virtualization

The idea of computer system virtualization is not new, the earliest example dates back to the 60s when IBM introduced the CP-40 and on 1965 with the IBM System/360-67 a 32 bit CPU with virtual memory hardware. However, in the last ten years, the use of virtualization in modern data centers increased. In 2001 and

2003 the first release of VMware server virtualization product and the first open-source x86 hypervizor, Xen. On 2005 free desktop virtualization.

Hypervisors are installed on server hardware whose only task is to run guest operating systems. In terms of virtualization approach there are three popular approaches to server virtualization: Full Virtualization, Para Virtualization and virtualization with hardware support (Hardware Virtual Machine, or HVM).

Full Virtualization provides emulation of the underlying platform on which a guest operating system and application set run without modifications and unaware that the platform is virtualized. This approach is idealistic, in real world scenarios virtualization comes with costs.

Providing a Full Virtualization implies that every platform device is emulated with enough details to permit the guest OS to manipulate them at their native level (such as register-level interfaces). Moreover, it allows administrators to create guests that use different operating systems. These guests have no knowledge about the host OS since they are not aware that the hardware they see is not real but emulated.

The guests, however, require real computing resources from the host, so they use a hypervisor to coordinate instructions to the CPU. The hypervisor provides virtual machine to show all the needed hardware to start and run the operating system.

Via a virtual bios, it shows CPU, RAM and storage devices. The main advantage of this paradigm concerns the ability to run virtual machines on all popular operating systems without requiring them to be modified since the emulated hardware is completely transparent.

Para Virtualization Virtual Machine approach is based on the host-guest paradigm and uses a virtual machine monitor. In this model the hypervisor modifies the guest operating system's code in order to run as a guest operating system in a specific virtual machine environment. Like virtual machines, Para Virtual Machines are able to run multiple operating systems. The main advantage of this approach is the execution speed always faster than Full Virtualization approach. The Para Virtualization method uses a hypervisor for shared access to the underlying hardware but integrates virtualization aware code into the OS itself. In a context of Para Virtualization the guest operating system must be aware of being run in a virtual environment. So the original operating system, in particular its kernel, is modified to run in a Para Virtualized environment.

Hardware Virtual Machine (HVM) Hardware-assisted virtualization is a virtualization approach that enables efficient Full Virtualization using help from hardware capabilities. Last innovations in hardware, mainly in CPU, MMU and memory components (notably the Intel VT-x and AMD-V architectures), provide direct platform-level architectural support for OS virtualization. For some hypervisors (like Xen and KVM) it is possible to recompile Para Virtualized drivers inside the guest machine running in HVM environment and load those drivers into the running kernel to achieve Para Virtualized I/O performance within an HVM guest.

1.2.2 Virtualization: some considerations

As anticipated, several advantages ensue from virtualization use. The reduced number of physical servers is one of the most evident: hardware solution for virtualization allows multiple virtual machines running on one physical machine, moreover additional advantages are on: power consumption reduction, better fault tolerance, optimization of time needed for device installation. Another advantage is that virtualization can be used for an abstraction of hardware resources, since operating systems are closely related to the underlying hardware, several issues need to be taken into account when a server is moved or cloned, e.g., incompatible hardware, different drivers and so on. Another virtualization feature is the creation of abstraction levels such that the operating system does not see the actual physical hardware, but a virtualized one. Administrator can move or clone a system on other machines that run the same virtualization environment without worrying about physical details (network and graphics cards, chipsets, etc.).

Virtualization allows resource allocation to virtual hardware easily and quickly. In some contexts it's necessary to maintain old servers with obsolete operating systems that cannot be moved to new servers as these OS would not be supported. In virtualized environments it is possible to run legacy systems allowing IT managers to get rid of old hardware no longer supported, and more prone to failure. In many cases it is appropriate to use virtualization to create a test environments. It frequently happens that production systems need to be changed without knowledge about consequences, i.e., installing an operating system upgrade or a particular service pack is not a risk-free. Virtualization allows immediate replication of virtual machines in order to run all necessary tests.

The two main open source hypervisor Xen and KVM based virtualization:
Xen is a Virtual Machine Monitor that allows several guest operating systems to be executed on the same computer hardware concurrently. A Xen system is structured with the Xen hypervisor as the lowest and most privileged layer. Above this layer

are located one or more guest operating systems, which the hypervisor schedules across the physical CPUs. Xen can work both in Para Virtualized or HVM mode; in the first the guest operating system must be modified to be executed. Through Para Virtualization, Xen can achieve very high performance. The HVM mode offers new instructions to support direct calls by a Para Virtualized guest/driver into the hypervisor, typically used for I/O or other so-called hypercalls.

KVM is a Full Virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). KVM is implemented as a module within the Linux kernel. A hypervisor hosts the virtual machine images as regular Linux processes, so that each virtual machine image can use all of the features of the Linux kernel, including hardware, security, storage, and applications. KVM supports I/O Para Virtualization using the so called VIRTIO subsystem consisting of 5 kernel modules IBM (2010).

1.2.3 Cloud services models

Cloud computing is a technology base on virtualization. The formal definition from The National Institute of Standards and Technology (NIST) is:

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models.

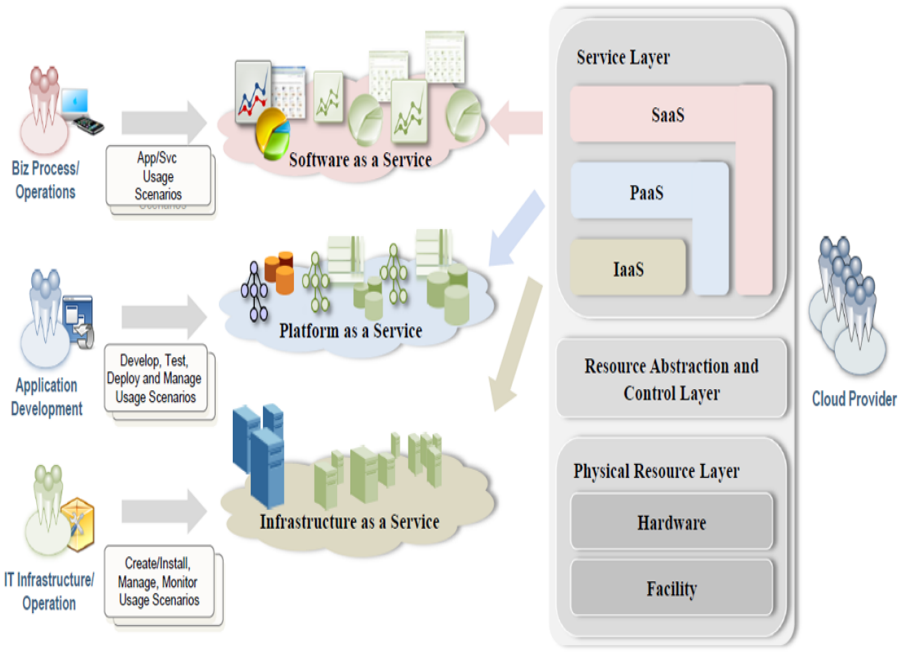

Figura 1.3. Cloud Computing Model

In order to be cloud main essential characteristics are:

- On-demand self-service: cloud users can unilaterally provision computing capabilities (disk space or computing cycles) without requiring human interaction
- Broad network access: users can access over the network through standard mechanisms using heterogeneous devices (e.g., mobile phones, laptops, and PDAs)
- Resource pooling: providers use a multi-tenant model to serve multiple users using the same pool of computing resources. Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.
- Rapid elasticity: resources can be elastically provisioned or released in order to rapidly scale the system. In some cases this can be done automatically.
- Measured Service: resources are periodically and automatically monitored and optimized providing transparency for both the provider and consumer of the utilized service.

Cloud computing is a concept (see fig. 1.4) that is to access resources, data and services on a remote server. Cloud computing is a technology that offers the possibility to create models of remote services, the existential currently 3 cloud services models.

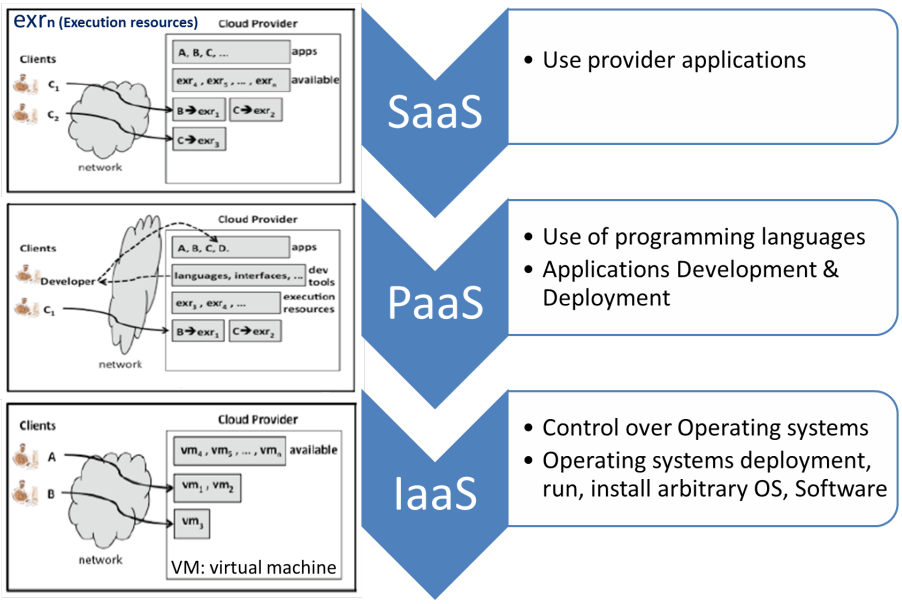

Figura 1.4. Saas, PaaS, Iaas

The **SaaS Software as a Service**: The capability provided to the consumer is to use the providers applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.(see Fig. 1.4).

The **PaaS Platform as a Service**: The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations. (see Fig. 1.4).

The **IaaS Infrastructure as a Service**: The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls). (see Fig. 1.4).

1.2.4 Cloud deployment models

Cloud computing systems was defined in 4 deployments models and 6 scenarios. The four models are:

- Private
- Public
- Community
- Hybrid Cloud

The six scenarios are:

- On-site Private Cloud
- Outsourced Private Cloud
- Public Cloud
- On-site Community Cloud
- Outsourced Community Cloud
- Hybrid Cloud

The on-site-Private cloud is a cloud Infrastructure solely for an organization. The private Cloud may be centralized inside the organization and at a single subscriber site (see Fig. 1.5).

The Outsourced-private cloud an outsourced private cloud has two security perimeters, one implemented by a cloud subscriber and one implemented by a provider. The two security perimeters are joined by a protected communications link (see Fig. 1.5).

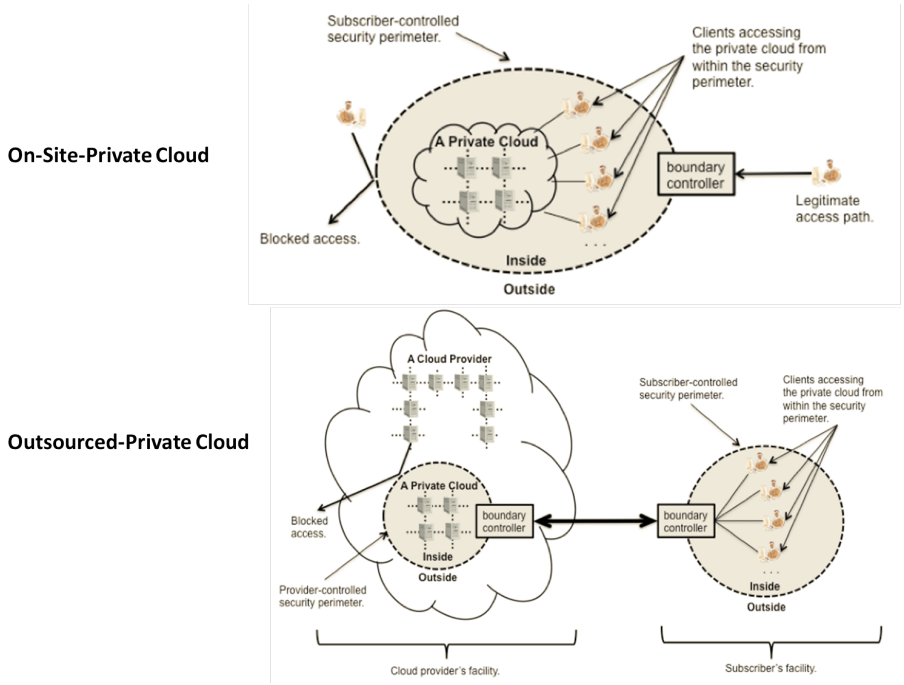

Figura 1.5. Private Cloud deployment

The on-site-Community cloud The community is a set of participant organizations. Each participant organization may provide cloud services, consume cloud services, or both. It is necessary for at least one community member to provide cloud services for a community cloud to be functional (see Fig. 1.6).

The Outsourced-Community cloud very similar to the outsourced private cloud scenario: server-side responsibilities are managed by a cloud provider that implements a security perimeter and that prevents mingling of community cloud resources with other cloud resources that are outside the provider-controlled security perimeter. The cloud provider may need to enforce a sharing policy among participant organizations in the community cloud (see Fig. 1.6).

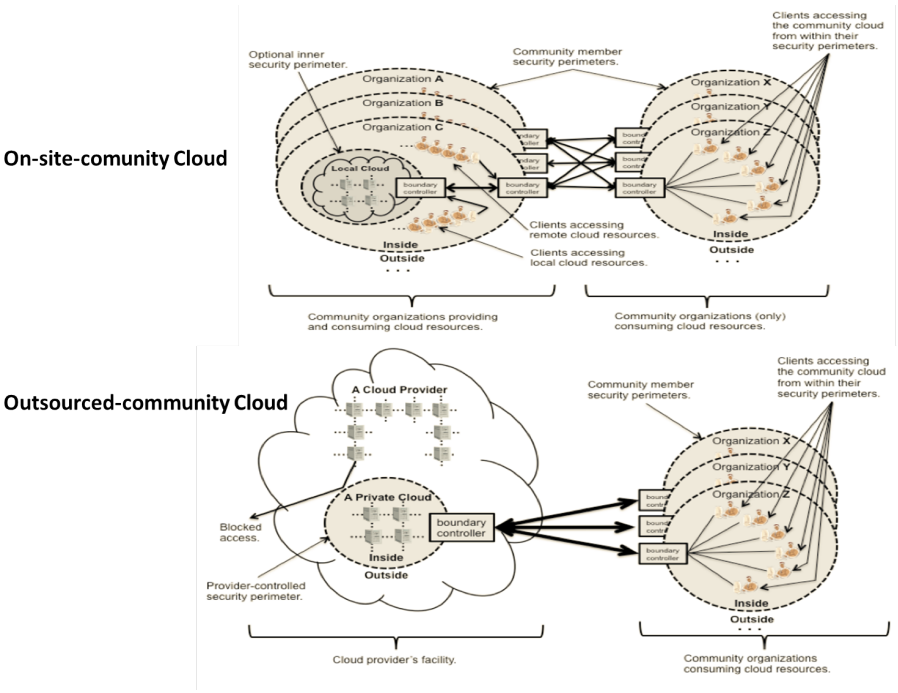

Figura 1.6. Community Cloud

The Public Cloud Resources are dynamically provisioned on a fine-grained, self-service basis over the Internet by a Cloud provider for single user or organizations (see Fig. 1.7).

The Hybrid Cloud Composition of two or more clouds (private, community, or public) that remain unique entities but are bound together, offering the benefits of multiple deployment models for the organization (see Fig. 1.7).

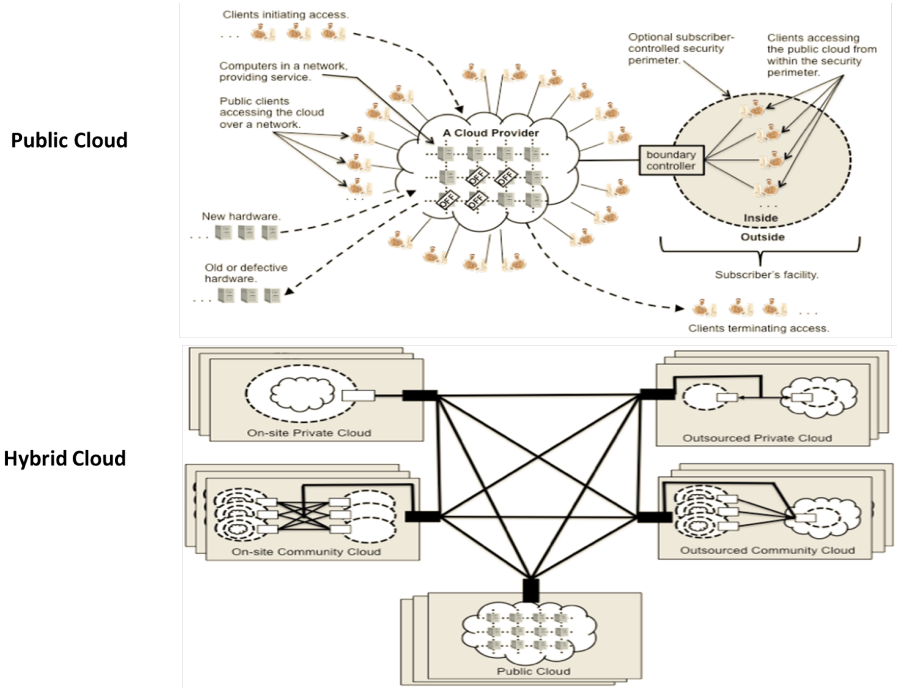

Figura 1.7. Public Hybrid deployment

Capitolo 2

ROSA ROSSA

2.1 The Rosa ROSSA project

In the years 2000-2001, the Italian Space Agency (ASI) started the development of a new GPS receiver devoted to Radio Occultation (RO) ROSA, Radio Occultation Sounder of the Atmosphere. The space-based GPS limb sounding, conventionally known as GPS Radio Occultation, is a relatively new remote sensing technique aiming to characterize the Earth atmosphere and ionosphere through the extraction of quasi vertical profiles of refractivity.

GPS Radio Occultation (RO) is an emerging remote sensing technique for profiling atmospheric parameters (mainly refractivity, but also pressure, temperature, humidity and electron density (Melbourne et al., 1994; Kursinski et al., 1997).

It is based on the inversion of L1 and L2 GPS signals collected by an ad hoc receiver placed on a Low Earth Orbit (LEO) platform, when the transmitter rises or sets beyond the Earth's limb. The relative movement of both satellites allows a quasi vertical atmospheric scan of the signal trajectory, and the profiles extracted are characterized by a high vertical resolution and a high accuracy.

The RO technique is applied for meteorological purposes (data collected by one LEO receiver placed at an altitude of 700 km produce about 300 or 400 profiles per day, worldwide distributed) since such observations can be easily assimilated into numerical weather prediction models. Anyway, it is also very useful for climatological purposes (the accuracy of inferred tropopause parameters is one of the most attractive aspects of the RO technique), gravity wave observations and space weather applications. Figure 2.1 sketches a concept view of the radio occultation technique, starting from the first operational RO mission on board the German

CHAMP satellite (Wickert et al., 2004), there are presently several other satellite missions carrying on board an RO payload. The most important are RO experiments on board the European METOP-1 mission (Luntama et al., 2008) and on board the USA/Taiwan COSMIC constellation mission. Several other missions are planned for the future.

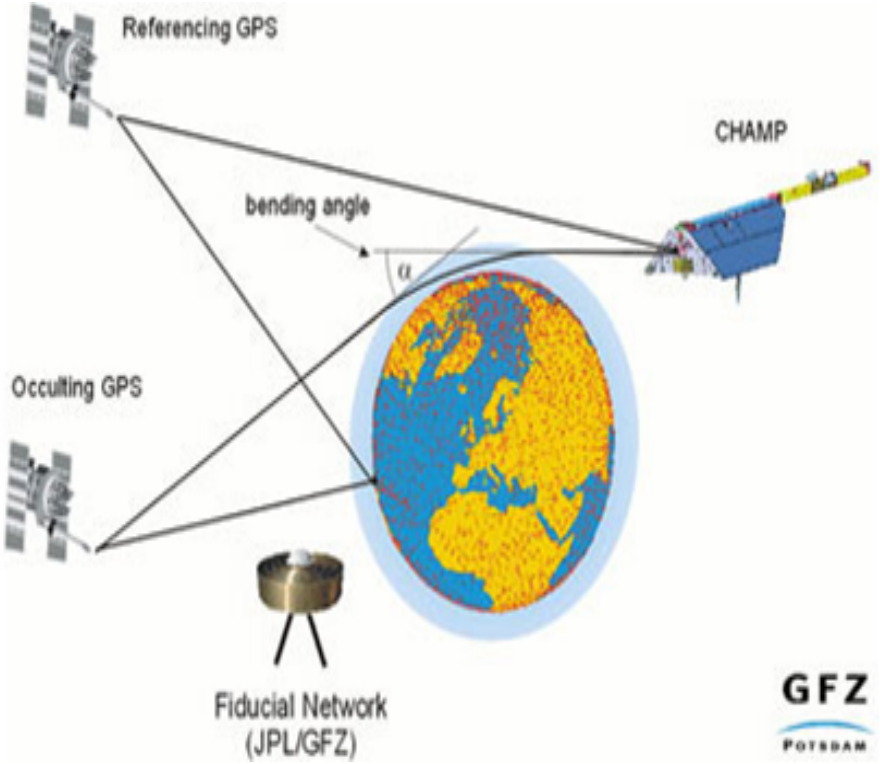

Figura 2.1. Radio Occultation: Champ satellite

In 2001, ASI (Italian Space Agency) has decided to launch a project called ROSA (Radio Occultation Sounder for Atmosphere) [9], consisting in the development of a GPS receiver for radio occultation and on board software and post-processing for the retrieval of atmospheric profiles. The GPS receiver and the onboard software has been developed by Thales Alenia Space, while, for the development of post-processing software, called ROSA-ROSSA (ROSA - Research and Operational Satellite and Software Activities), was founded a research group joining several Italian universities, including the Remote Sensing group of the Politecnico di Torino. The receiver ROSA was sent into orbit Sept. 23, 2010 aboard the Indian space mission OCEANSAT-2, and put into operation after two days.

He is currently fully operational [8]. The ROSA ROSSA software was designed in two versions called: Base Version (VB) and Extended Version (VE) [13]. The first involves the retrieval of altitude profiles of bending angle, refractive index, pressure, humidity, temperature and electron density through the use of geometrical optics, while the second involves the improvement of the profiles in the lower troposphere through the use of the resolution of multipath techniques.

The use of the technique of GPS Radio Occultation is able to study in detail the atmosphere to obtain physical parameters. The main scientific applications of the Radio Occultation technique are in the field of: Meteorology, Climatology, Ionospheric Studies, physics of the solid earth. The core activities of the ASI has been the development of the ROSA receiver. This is able to realize vertical profiles of temperature, pressure and humidity of the atmosphere with high vertical resolution (about 200m and 1000m in the troposphere into the stratosphere) and with high accuracy in the measures of temperature. The main purpose of this driven project is to combine the algorithms of Radio Occultation with Grid Computing, in order to have one part of a collaboration between the scientific community and various organizations on the other hand, speeding up data processing necessary for radio occultation.

The ROSA project has also initiated various scientific and technological activities related to the space segment, ground segment and to the users for the study of the technique of radio occultation. The main initiatives for this mission are: Development and construction of a tool for GNSS Radio Occultation of satellites currently operating Installation of ROSA on various space missions Development based on ASI Multi-Mission Center for the Development of advanced processors Radio Occultation data Implementing an open data policy.

The products obtained by processing radio occultation data can contribute significantly to many fields of research. In particular: Climatology: the long-term stability of the instrument ROSA can ensure continuous monitoring of the atmosphere for a long period of time, giving relief to the climatological observations themselves. In particular on:

- Meteorology: Weather on a global scale, with expected improvements due to the acquisition of weather data in areas for which no data are available on the ground (oceans, polar regions, deserts). The occultation data can contribute to global knowledge of moisture distribution, the monitoring of the tropopause and reconstruction of the temperature profiles on the lower stratosphere. RO data assimilation in NWP (Numerical Weather Prediction).

- Space Weather: Physics of the ionosphere, with expected improvements in research and forecasting of ionospheric time. Dynamic aspects of the ionospheric plasma can be monitored by measuring scintillation, while the continuous monitoring of the overall density of electrons can provide the database needed to test the predictive models.
- Space Geodesy and Geophysics: studies of the Earth's gravitational field, for the investigation of the solid earth science. In fact, the determination of orbits can be improved through better understanding of the atmosphere itself. Emerging applications related to the use of a GPS receiver in a bistatic configuration can allow, through scatterometry measures and elevations the study of sea state and properties of ground.

2.2 Scientific Context: GPS Radio Occultation

The technique of Radio Occultation GPS is a remote sensing technique intended to characterize the atmosphere and the terrestrial ionosphere, through the reversal of measures of excess phase acquired from a GPS receiver on board a satellite LEO (Low Earth Orbit) in occultation compared to the radio transmitter.

Radio occultation is refers to a situation in which a transmitter and a receiver are in condition to communicate even if they are not in an optical view, ie, any object placed in the them, prevents them from seeing. If the signal propagation between transmitter and receiver is in free space, and then with rectilinear propagation, neglecting the possible effects of diffraction, an object placed between the two, would block all of the communication.

In general, the observable on which is based the technique of radio occultation for the retrieval of atmospheric profiles are the amplitude and the phase delay accumulated by the GPS signal in its propagation through the Earth's atmosphere. A schematic representation of the geometry of an event of radio occultation and of the main parameters involved is reported in Figure 2.2.

From the physical point of view, it's the analysis of what happens to a signal transmitted by a satellite in the GPS constellation and received from another satellite placed in low earth orbit (LEO Low Earth Orbit to about hL = 400km). The phenomenon of concealment radio occurs when the GPS satellite rises (event laughed) or sunset (event set) with respect to the satellite in LEO, so that the trajectory followed by the signal intersects the various layers of the atmosphere. The wavelength of GPS signals is of the order 20cm, much less to the scales of spatial variation of atmospheric variables measured. Consequently, the traditional method of retrieval of atmospheric profiles is based on the application of the theory of geometrical optics, that allows to describe the propagation of any electromagnetic wave in terms of ray. Using this approximation, a ray that passes through the atmosphere comes from this refracted in accordance with Snell's Law. As the observable fundamental of this technique essentially a measure of the phase delay suffered by the signal, the presence of the atmosphere will influence him with two distinct contributions: the first, of geometric type, caused by the curvature of the trajectory (the signal is propagate along a path longer than that geometrically straight conceivable in a vacuum) and the second, of the optical type, caused by a propagation velocity less than the speed of light (the refractive index is in fact slightly higher unit). The trajectory will be the one for which it is verified Fermat's principle, namely, that for which is the minimum phase delay suffered by the total signal.

When the path begins to intersect the upper atmosphere terrestrial (with a

perigee located near the mesopause, about 85km of share), this will begin to affect the phase delay which, if expressed in terms of wavelength equivalent, assumes values close to the millimeter. As concealment of the event progresses, the trajectory will intersect zones of the atmosphere denser and denser, until the radio link between the two satellites involved is interrupted. The phase delay at this point will be incremented until to take values close to km. Ultimately, the Earth's atmosphere generates a observable with a dynamic range of about 6 orders of magnitude.

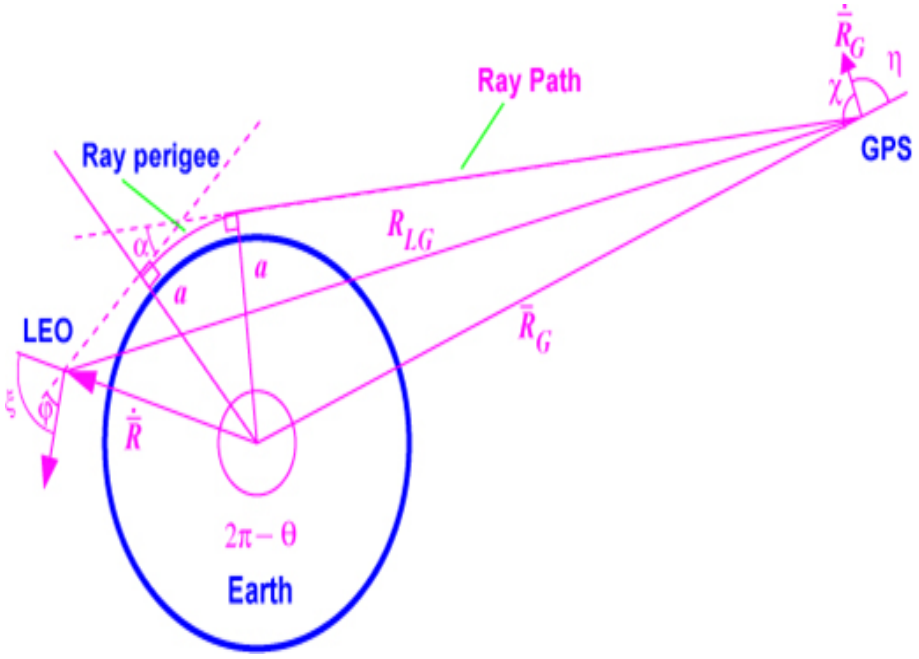

Figura 2.2. Radio Occultation technique

More different situation occurs on the radio occultation by satellite. In this case, the electromagnetic signal emitted by a transmitter placed on the satellite, through the Earth's atmosphere and is received by a receiver located on another satellite in low orbit. The word occultation implies a geometry the type named transmitter, a planet with its atmosphere which acts as a barrier, and a receiver in movement (see figure 2.3).

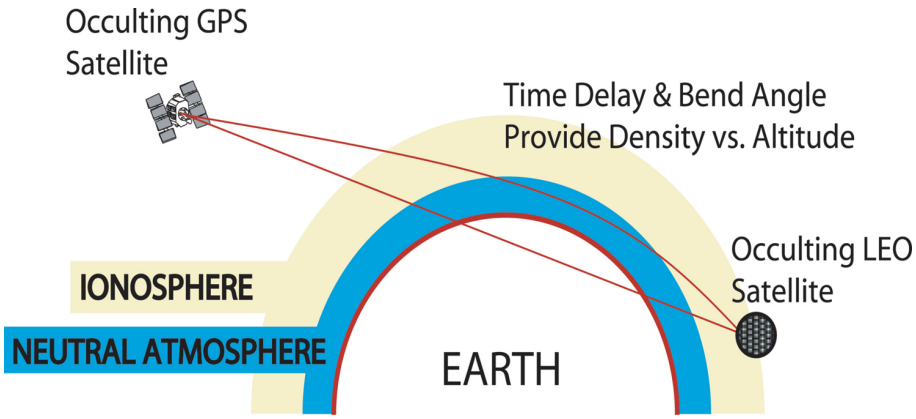

Figura 2.3. Occultation: GPS and LEO satellites

Sometimes, reference is made in an improper manner even in situations type transmissions of signals between satellites for the study of the ionosphere, or reception of signals reflected from surfaces, but such situations, always include a planet with its atmosphere as a transmission medium. Respectively to the direction of movement of the receiver, and looking from the point of view of the transmitter, an event of radio occultation can be of the rising if the receiver is located relative to the horizon terrestrial, or setting if the receiver sets with respect to 'Earth's horizon.

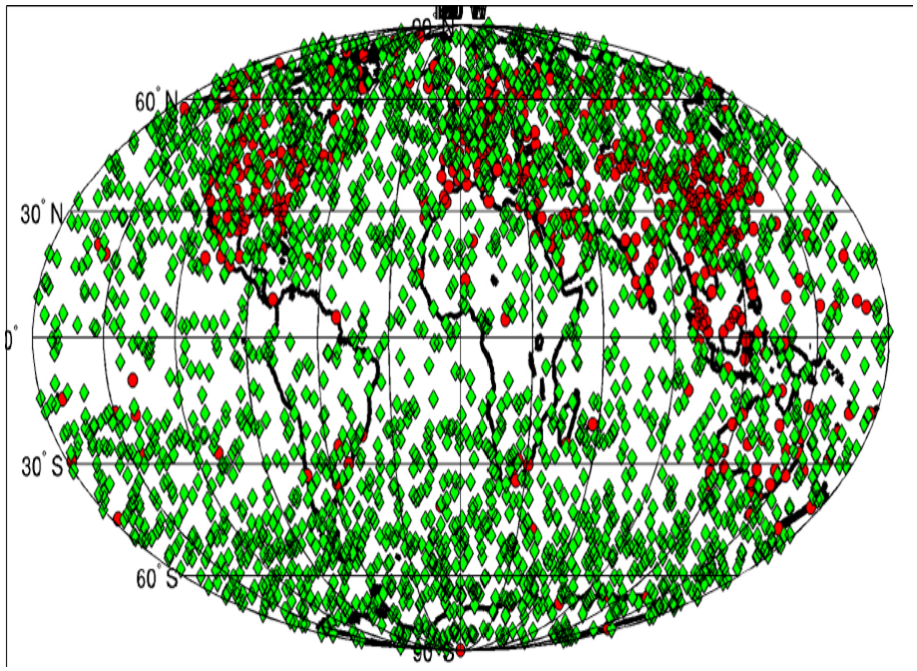

Figura 2.4. Cosmic Space Mission

When the signal transmitted from the satellite is propagated in the Earth atmosphere, its speed and direction of propagation will vary depending on certain atmospheric parameters. The phase and amplitude of the signal are then altered, and their values compared to the values that would be obtained in straight propagation (in the vacuum), are useful to determine the characteristics of atmospheric refractivity crossed. On figure 2.4 Daily Radio occultation events distribution from six satellites on the COSMIC space mission.

2.3 Data elaboration

The ROSA-ROSSA extended version software implements state-of-the-art RO algorithms which were already available from the scientific group and are used during the validation phase before their final transfer inside the official ground segment of the ROSA radio occultation receiver. The processing chain, which is subdivided into seven different software modules (Data Generators (DGs), are executed in a sequential mode. Figure 2.5 shows a simple diagram of the processing chain and the corresponding dataflow.

Independently of the type of release Base Version or Extended Version, the software-ROSA ROSSA consists of different modules, called Data Generators (DG) [10], executed in a sequential manner. Each module receives input data from a previous module and produces the other to be given as input to the next module, until obtaining the final result. After the SWORD Orbital Module determination, the hull is divided into two subchain, the first terminating in the form DG ATMO, the second containing only the module DG_DELN. The following describes in more detail the individual modules.

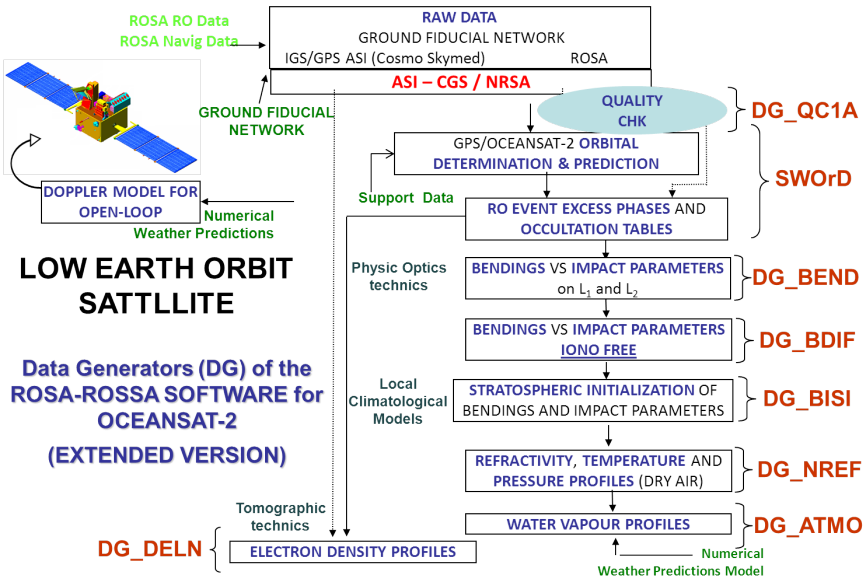

Figura 2.5. Data Generators Chain

- SWOrD (Software for Orbit Determination) is the first data generator carries out the calculation of the LEO Low Earth Orbit satellite

and excess phase on both carriers L1 and L2. For the calculation of the orbits, the base version uses data from IGS (International GNSS Service), while the extended version made estimation through extrapolation algorithms. The excess phase and the orbital parameters are input data to the two modules DG_BEND and DG_DELN. SWOrD is a software module that fully supports orbit determination, orbit prediction, and implements data generation connected with the ROSA sensor on board OCEANSAT-2. Input data for SWOrD are ROSA GPS navigation and radio occultation observations, ground GPS network data and other support data.

It generates the following output data: estimated rapid orbits and predicted orbits for GPS Data intensive scientific analysis with grid computing constellation in the conventional terrestrial reference frame (data level 1b.a); 50 Hz closed-loop and 100 Hz open-loop excess phases and signal amplitudes data for each single occultation event; tables showing estimated and predicted (up to 6 hours in advance) occultation.

- DG BMDL predicts a bending angle and impact parameter profile usable as input in the ROSA on board software excess doppler prediction module for open-loop tracking. For each predicted occultation event, the latitude and longitude of geometrical tangent points (the nearest point of each trajectory to the Earth's surface, evaluated through predicted orbits) is used to compute the bending angle and impact parameter profile from interpolated numerical weather prediction models (the bending angle and the impact parameter are geometrical parameters univocally identifying each trajectory followed by the RO signal. The predicted bending angle and impact parameter profiles are stored in ASCII data files containing bending angles and impact parameters together with the UTC time stamp, one file for each event. Input data for DG BMDL are predicted GPS, LEO orbits and predicted occultation tables, together with ECMWF world forecasts for synoptic times valid for a future observed occultation event.

- DG_BEND This Data Generator carries out the calculation of the profiles of bending angle and impact parameter bearing on both L1 and L2 from the values of excess phase received from SWORD. These profiles are still suffering from error due to the passage of signals through the ionosphere. In the basic version (VB) the approximation of geometrical optics are used, while the extended version use the Full Spectrum Inversion (FSI).

- DG_BDIF This Data Generator made a compensation in the ionosphere, eliminating errors in the calculation of bending angle and impact parameter due to the passage of signals through the ionosphere. The compensation is performed using a linear combination of values of bending angle and impact parameter of both the carriers, in order to obtain a single profile valid for both the carriers, as if both signals do not have a perturbation of additional delays due to their passage through the ionosphere.
- DG_BISI This Data Generator performs an optimization of the statistical profiles of bending angle and impact parameter in the stratosphere (above 40km) using climatological data (ECMWF or in VE). The optimization is accomplished with a statistical combination of the bending angle output from DG_BDIF and the same obtained by climatological data (ECMWF for VE), in order to obtain the more attending bending angle. The algorithm assigns more weight to the climatological profile if the standard deviation of the measured profile is largest than the standard deviation of the climatological profile and vice versa. Consequently, the contribution of the bending angle climatological increases with increasing altitude.
- DG_NREF This Data Generator receives as input the profiles of bending angle and impact parameter output from DG_BISI cleaned by ionospheric errors and offset in the stratosphere and calculates the profiles of refractivity, dry temperature and pressure through the inversion of the integral of Abel. DG_NREF (REF for refractivity) this data Generator provides the refractivity, the dry air temperature and pressure profiles. This Data Generator processes iono-free and initialized bending and impact parameter profiles in order to compute the correspondent quasi vertical refractivity profile, by inverting bending and impact parameter profiles trough the classical inverse Abel integral [see Melbourne et al., 1994]. In addition, pressure profiles will be available after this processing stage by integrating the hydrostatic equation (starting from climatological pressures relative to higher levels). Dry temperature profile extraction is straightforward. The geodetic latitude and longitude correspondent to the location of each point of the profile (which corresponds to each trajectory perigee) is given on a global domain above the EGM96 geoid model. An ASCII file will be available on output.
- DG_ATMO This Data Generator carries out the calculation of the profiles of water vapor and temperature using climatological data for the Base version and data NWP (Numerical Weather Prediction

date) for the Extended Version. This Data Generator allows the evaluation of water vapour and temperature profiles by using Global Climatological Models (in the first release) or Numerical Weather Prediction data (in the final release), following a 1DVAR approach. allows evaluating the temperature and water vapour profiles using forecasts or analysis obtained by numerical weather prediction. This DG receives input data files and produces an output data files, which contain the total temperature and total pressure profiles in terms of wet and dry components.

- DG_DELN This Data Generator computes the electron density profile in ionosphere. It receives on input three files: GPS orbit, LEO orbit and L1 and L2 excess phase data collected from the occulted satellite. Each file corresponds to one occultation event. It produces electron density vertical profiles following the onion peeling approach [see Melbourne et al., 1994]. For each event, an ASCII file is produced on output. Locations of each profile point (latitude, longitude and altitude) are also stored.

2.4 Technical and computing requirements

An objective, in terms of requirements, was to merge RO algorithms and Grid Computing in order to apply, on one side, a collaborative engineering approach and, on the other, to reduce the time required by the overall data processing. For what concerns collaborative engineering aspects, it has to be noted that the Grid Computing environment has been chosen since what we had to develop and to made quickly available (given the forthcoming OCEANSAT-2 launch) through modular software [5, 6].

By using a Grid Environment it has been possible for each partner (responsible for each software Data Generator module) to develop his own module independently of the other partners, by simply exploiting the distributed hardware and software environment and using on input the data from other missions, made available after a reformatting procedure at the various ROSA-ROSSA data levels.

Thanks to the availability of such distributed and not operative ground segment, in the future it will be possible for each partner to manage its own data generator, improving it if necessary, testing it and easily replacing in the overall operational processing chain. More important, it will be possible for other interested research groups distributed all over the world to create their own software module working with the ROSA data at a particular level, and to introduce in the overall processing

chain by simply respecting well defined interfaces.

This will be one of the main strength of the distributed infrastructure and software environment with respect to the classical centralized Radio Occultation ground segments, which are generally concentrated in well defined data center and whose software modules are not easily to be upgrade.

The second strength point is the possibility to significantly reduce the processing time thanks to the high number of distributed workstations [6] and consequently to use the global computational capacity. When a user wants to process a collection of data, more than one job could be sent and processed through the Distributed Architecture. In this case we could obtain results in less time due to the opportunity to use more than one node for each execution. On a grid environment it's easy to obtain an overall reduction of time necessary to process the radio Occultation events by parallelize process jobs, with respect to the processing time observed in a centralized infrastructure by running sequential jobs in a single workstation.

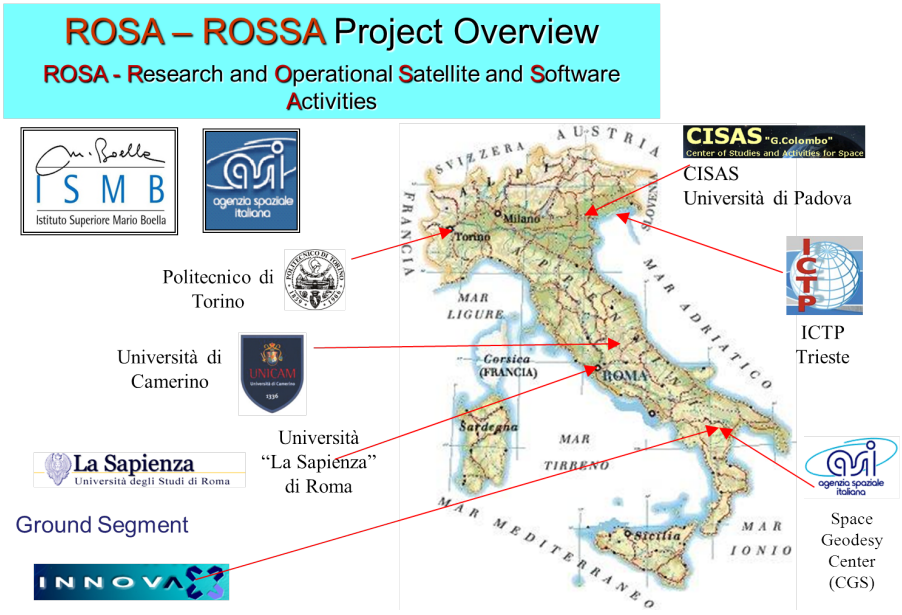

Figura 2.6. Web science

Moreover, parallelization mechanisms exploitable by using the Grid approach will be adopted in the final ROSA-ROSSA release, in particular for what concerns the GPS and LEO orbit determination. Both these aspects will be particularly important in the future, when raw data are provided to the ROSA-ROSSA processing chain

within the real time constraints (is not the case for ROSA on-board OCEANSAT-2). This will allow the ROSA-ROSSA software to make Radio Occultation products available to the meteorological center for weather prediction applications, as already is done by the other RO processing center.

In term of data transfers a large amount of data is sending for submitting jobs from several different organizations (Italian Universities and Research Centers) located in many different places (see Fig. 2.6)

For scientist community grid services are available for creating a user friendly environment. A specific open source middleware is installed on each nodes inside the Grid infrastructure. For Grid ROSA the Globus Toolkit was used. It is a fundamental enabling technology for the Grid, its task is allow sharing computing power, databases, and other tools securely online [Foster et al., 2003; Berman et al., 2005]. The Globus middleware toolkit [www.globus.org/toolkit] includes software for security, information infrastructure, resource management, data management, communication, fault detection and portability. It is packaged as a set of components that can be used either independently or together to develop applications. The Globus Toolkit was conceived to remove obstacles that prevent seamless collaboration. Its core services, interfaces and protocols allow users to access remote resources as if they were located within their own machine room, while simultaneously preserving local control over who can use resources and when.

2.5 ROSA on Hybrid infrastructure

2.5.1 Infrastructure Design

The rosa Grid Processing Infrastructure [3] is an integrated system devoted to handle and process RO (Radio Occultation) data of the OCEANSAT-2 ROSA on board sensor. The rosa Grid is managed by the Grid Processing Management (GPM) and is composed of the following subsystems:

- A middleware Globus for grid services
- A central repository: responsible for storing data in directories and files; it stores executable files, source, input/output and support files.
- A relational database: for storing data about node description, grid configuration, Data Generators description, all input files needed for execution of each DG, user account and credentials, logs for jobs status;
- A Java Application: providing a graphical interface used to submit jobs from each node
- Jobs scheduler: responsible for the identification of available resources on the Rosa Grid infrastructure and for submitting the processes.
- Data Generators application: a sets of softwares algorithm modules (compiled matlab) for processing Radio Occultation files.

As explained on previous section the general purpose of the project was to share the computational resources [7, 11], transferring a great amount of files and submitting jobs from several different organizations of the scientific community located in different areas of Italy. The nodes are located geographically in Italy, namely, at:

- Istituto Superiore Mario Boella (Turin)
- Polytechnic University of Turin (Turin)
- University of Padua (Padua)
- Sapienza University (Rome)
- University of Camerino (Macerata)
- International Center of Theoretical Physics (Trieste)
- Italian Space Agency (Matera)
- Institute for Complex Systems (Florence)
- Innova Industrial Partner (Matera)

Hence, the Globus Toolkit as middleware (Berman, 2003; Globus, 2010a; 2010b) is used for grid services, since it allows obtaining a reliable information technology infrastructure that enables integrated, collaborative use of computers, networks and databases. The configuration of the Grid used for the project is composed of 10 machines: a master for Grid management architecture, a backup node, a repository located in Turin, Italy. We used platforms of 64 bit, Dual e Quad Core capabilities,processor is higher than 1.6 GH and at least of 2 Gb of RAM. - seven client nodes, geographically located in several Universities and Research Centers in Italy see figure 2.7 were actived.

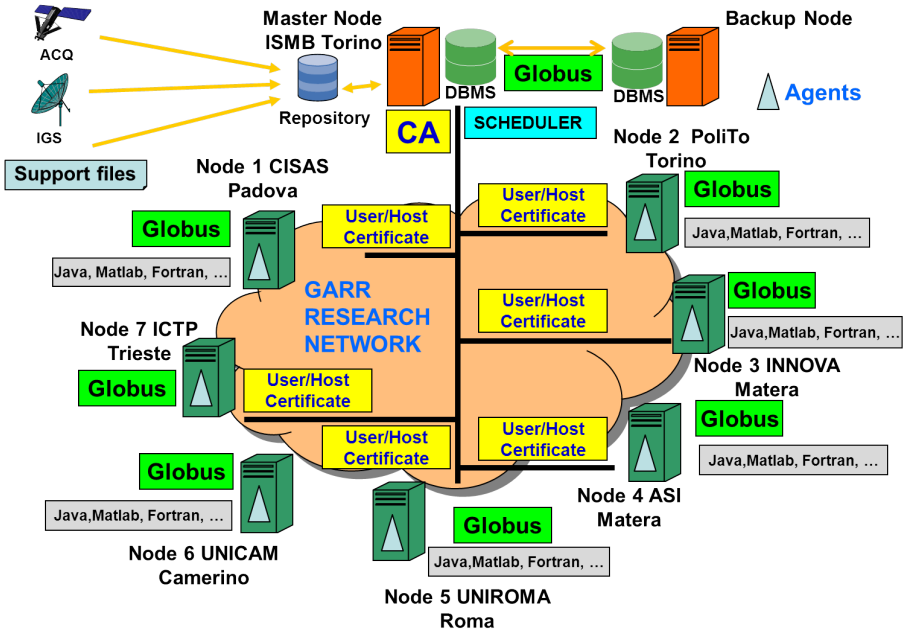

Figura 2.7. Grid Rosa infrastructure

The main goal of the master node is to monitor the Grid Resources, but it also handles the storage of output files in a centralized repository. The master node contains information about every resource belonging to the Grid. Instead, the Backup Node is a replication of the master node and it works only if master node has some troubles or it is down. The client nodes can submit job requests through a stand alone Java application [http://wiki.cogkit.org]. Users can send request of job execution in two different ways: a single job request or a processing chain for a specific event.

2.5.2 Data Management Model

The data model, based on a MySQL Database Management System, is a fundamental component of the Rosa Grid Infrastructure. It has three main functions: one for grid configuration and user accounting, one for Data Generators algorithm description, and one for data logs. The data model is composed of six domains in order to define the main function also named entities.

GRID CONFIGURATION: It is dedicated to the nodes' description, the technical characteristics of each node, the path where are stored files and the network parameters such as IP, MAC address and location. This information is useful for the scheduler and middleware.

DATA GENERATORS DESCRIPTION: This set of tables describe each data generator. Each table is deputed to store information about files, softwares and libraries list needed for its execution, the path where all sources and support files can be found. All DGs are characterized by an XML file, and we also describe the XML file structure for each DG; in this way, we are able to dynamically generate the XML file for each event.

GRID STATUS: This set of tables are dedicated to the scheduler; each table contains all information concerning the general status of each node and how many services are up/down.

GRID STATISTIC: This set of tables store information about the behavior of some parameters (CPU and RAM) on each node and the total processing time for each DG on each node. A policy rule for the scheduler is based on the estimation of the total elaboration time.

USER ACCOUNT AND GRID CREDENTIAL: On the grid, each user has different permission, such as executing DGs, reading data and showing output.

2.5.3 Scheduling model

ROSA Data Generators applications have several programming languages involved in the processing chain and a high number of input files [Hongzhang et al., 2004]. A specific ad-hoc scheduler was developed in order to takes some the following fundamental aspects:

- nodes capabilities: software and specific library installed, cpu and ram memory;

- job requirements: software, library, process time estimation needed.
- job scheduling: in order to select the best node to run a job, the scheduler assigns to each node a Unit Capacity (Token) depending on specific weight criteria like: CPU usage and available RAM.

Users can ask to submit a job on the grid through (see fig: 2.8) by a Java application (1,2), the scheduler queries the DB on Master Node (3), the scheduler through scheduling policy selects which machine has the best performance for job execution, assigns the job to selected node (4), each node through an agent sends information about its own status to the GPM master node. The scheduler provides for automated scheduling of any input files, processes as well as import and export of data (Buyya et al., 2000). It uses all machines belonging to the grid to distribute the work load and provide a backup system for all critical tasks within the system (Kurowski et al., 2006). It has the ability to run more than one job at the same time and run multiple steps within a single job in parallel to complete jobs faster and more efficiently (Leonid, 2004). Its tasks are all created as Java class files, allowing the creation of new and complex tasks in an easy manner. Tasks can also be run with a configuration parameter if desired, creating only a part of the chain.

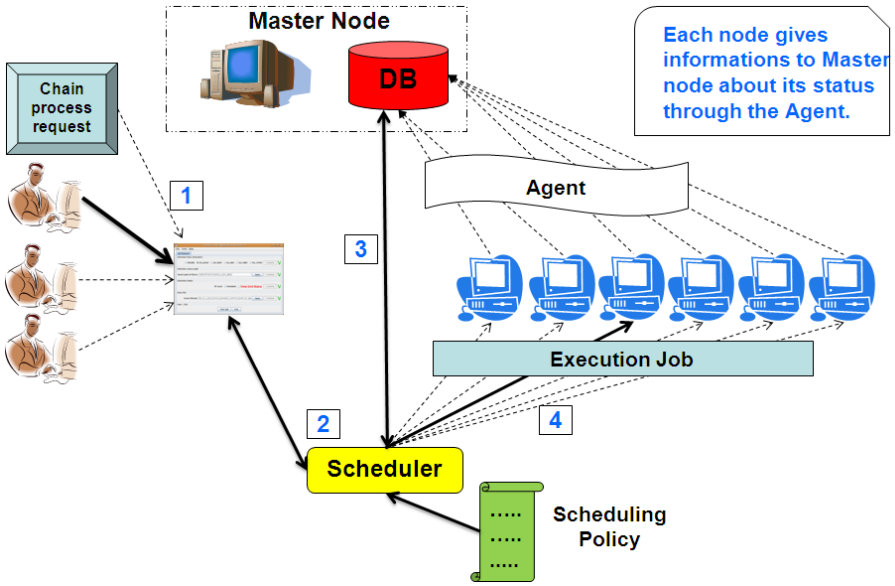

Figura 2.8. Grid scheduling

For the purpose of the project Two schedulers were developed [4]: a global and a local scheduler. The global scheduler is installed only on the master node and makes

it possible to check the number of files to be processed and the number of available nodes, and send files to these machines. This scheduler decides how to share the computer among the processes that are ready to run. The choice of how to share the file to run is based on two sets of scheduling rules, one concerning the available nodes and one derived from an analysis of the file to run.

The **global scheduler (GS)** is split in modules, one module for each data generator. Each module is a daemon and is in charge of the scheduling of jobs. Each module is independent. Six modules have been implemented, each of them based on Java implementation from the Globus Toolkit. Due to the requirements of RO analysis, a job is defined by the presence of data files to be analyzed. Each module checks the availability of files on a specific directory, one for each Data Generator. The DG module is responsible for getting the list of files for analysis.

The next step is to check the availability of the node by querying the grid status table on the data model. A list of available nodes is returned; for each node the schedule module checks if it is in condition to run a job by sending a globus command. The last condition is to check how many nodes are running; the module sends files by choosing free nodes. If all nodes are running, the files are in a queue condition. Finally, each node has a threshold for its computational capacity and, according to this criterion, one or more files are sent for the analysis.

The **local scheduler (LS)** is a daemon and has in charge the run execution of each DG. Each node has a set of directories that correspond to each repository of files for each DG. The local scheduler is implemented with a round robin policy in order to give a rotation for the elaboration of each DG algorithm. The local scheduler takes the list of files and for each of them runs the correct DG algorithm. DG algorithms are developed in a Matlab compiled file. When the execution job is accomplished, the output file is sent to the master node and is located in the next DG directory. In this case, it will be managed by the global scheduler for the next elaboration.

As explained above, the whole processing chain [13] is composed of seven different Data Generators executable. These parameters are retrieved after executing of complex algorithms which involve a set of languages like Fortran, MatLab, C++, Mathematica and Java. The usage of this heterogeneous mix of technologies increases the scheduling policy implementation complexity. In a typical scenario, the job chain is executed by a single machine, in which different events run one by one: in case of N events to be executed, event N can be processed only when the previous N-1 events have been completed. This sequential architecture is defined as Non-Distributed Architecture (NDA).

In our scenario, since each event is split into six sub-activities corresponding to as many algorithms, we can exploit the Distributed Architecture (DA) in order to benefit from job parallelization (see Fig. 2.9). In this way we are able to submit jobs, referred to several events, to different nodes at the same time, thus reducing the time required for determining the atmospheric parameters referred to each event. In both examples the start time is set to t_0, but while in the DA event execution terminates at $t_0+\Delta_t$, in NDA it finishes at $t_0+\alpha_t$, with $\Delta_t < \alpha_t$. This time reduction is attributable to the fact that in the DA it is not necessary to execute all the jobs related to one event in the same node. It is instead possible to run, for example, in Node 1 the BEND job from Event1 but also in the same time the BEND job from Event 2 on Node 2.
In a non-distributed architecture the time of execution is:

$$T_p = Ts + (Te * \eta)$$

In a distributed architecture the time of execution is:

$$T_p = (Ts + \beta_s) + \frac{(Te + \beta) * \eta}{N}$$

Where:

$$T_p = TotalTimeProcess\ T_s = SwordTimeProcess$$
$$T_e = EventTimeProcess\ \eta = ROEventNumber$$
$$N = GridNodeNumber\ \beta = TotalFileTransfertime$$
$$\beta_s = SwordFileTransfertime$$

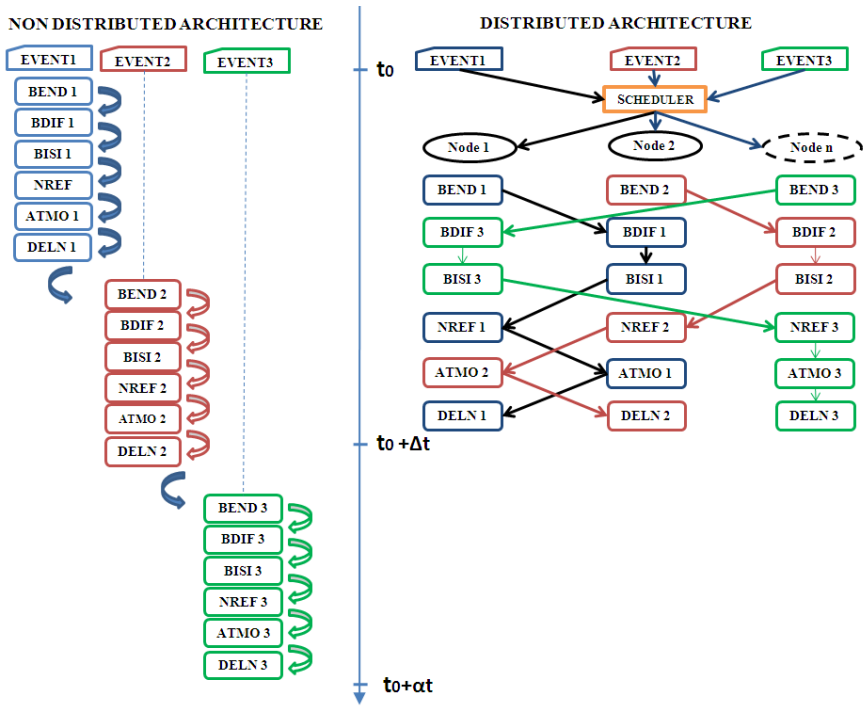

Figura 2.9. Distributed and Non distributed Time processing

In phase of analysis specific attention was dedicated to the queue approach and how to manage the queue of jobs request. In term of length of execution time for each DGs a learning phase from all DGs over all nodes. In this way a clear indication of the process duration time for a specific DG on each node is available. This is an important feature for the RO scenario due to the panel of nodes cover Dual Core and Quad Core capabilities. In fact, considering the large amount of events, estimated 256 by day, and for each event the characteristic of the chain process, it is not so unfrequent that all nodes are in running condition. In this way a minimization for the end time of jobs running on each machine was possible, the start time was stored, the duration of estimated process time for each jobs (DG) on each worker nodes was stored and the selection of the node that finishes first his running job was stored by inserting the new job requested in queue.

2.5.4 Grid infrastructure monitoring by agent

In spite of the success of grid computing in providing solutions for a number of large-scale science fields, one of the problems is the scalability of the system. The agents are emerging as a solution to provide flexibility and scalability (Gradwell,

2002). The agent is installed on each node, is used to monitor the availability of each service on the node and periodically sends its general status to the database on the master node.

In a specific node all services must be available and ready, this is a condition for receiving and execute jobs. The scheduler, for the selection of available nodes ready to run, instead of querying each machine, queries the database directly. A specific function checks if the feedback information from all nodes has been sent and, in the case of a missing status, the node is considered unavailable for processing data. The agents solution was choose for providing flexible, autonomous and intelligent software components [Kurowski et al., 2006]. On the architecture, there are two type of agents:

- Jobs Agent Monitoring, to monitor some parameters on nodes during jobs execution for getting status of jobs (running, waiting, completed, failed);
- Resources System Agent, to monitor services status on nodes for giving a feedback of the general status of the nodes in term of availability and reliability.

One of goal was scalability. Furthermore, for jobs request, the scheduler never queries nodes on grid to get information, but it only queries the relational database. The main objective of the agent was decentralization so the solution is extremely scalable; all parameters needed for the scheduler are sent by the agents from each nodes and stored on database.

2.5.5 Automatic processing chain

The software modules developed and the infrastructure allows to run the processing chain automatically; it is composed from two schedulers: one on the master node, called global scheduler, and checks for jobs ready for execution and sends them to worker nodes, according to the scheduling rules, the other, called local scheduler, installed on each worker nodes returns the results for each output file obtained on the master node.

The first transaction takes place on master node; it receives the files directly from the ground station and performs the first step in the chain, i.e., SWOrD, generating about 256 files that are placed in the folder the next step, DG BEND. When files are available in the folder DG BEND, the global scheduler checks nodes available by querying the database, and sends files to them. Global scheduler provides for automated scheduling of any input files. It uses all machines belonging to the grid to distribute work load and to provide a backup system for all critical tasks within the system. The choice of how to share the file to run is based on 2 sets of scheduling rules, one concerning the available nodes and one derived from an analysis of the file to run.

An agent is installed on each node, is used to monitor the availability of each service on the node and periodically, it sends its general status to the database on master node, if all services are active the node is in condition to receive a job. For the selection of available nodes and ready to run jobs, the global scheduler checks on the database directly instead of querying each machine. Two types of errors can occur: the first for lack of data in the file due to the satellite reception, the second for network failures or node crash.

Only in the last case it is worth recover the process, and it is enough reprocess it. Anyway, each process has a timeout, if within a fixed time processing has not been completed (fixed time obtained due to a learning phase for each DG), the process is killed, a specific software module was developed for this feature. An important component of this architecture is the database, which allows us to monitor any action of the grid. Regarding the automatic chain, each transaction is stored on the database when it starts running, when it ends, for input and output files involved, the node that has run and type of error, if it has generated them. The database also contains information on the status of each node and are available to receive the file to run, this allows to understand whether there are network problems, so if the node is reachable.

When workers nodes sees a file in its folder (see Fig. 2.10), it starts the processing procedure that will generate an output file that will be sent on the master node in

the folder next step, i.e., DG BDIF. This procedure is performed for each step of the chain, and the operation is as follows: from SWOrD, the DG n-1 generates the output file that will be the input files of DG n, and so on. On worker nodes, each execution is performed in a temporary folder so that, in the case of an error, it is possible to identify the type of error made and then reprocess jobs.

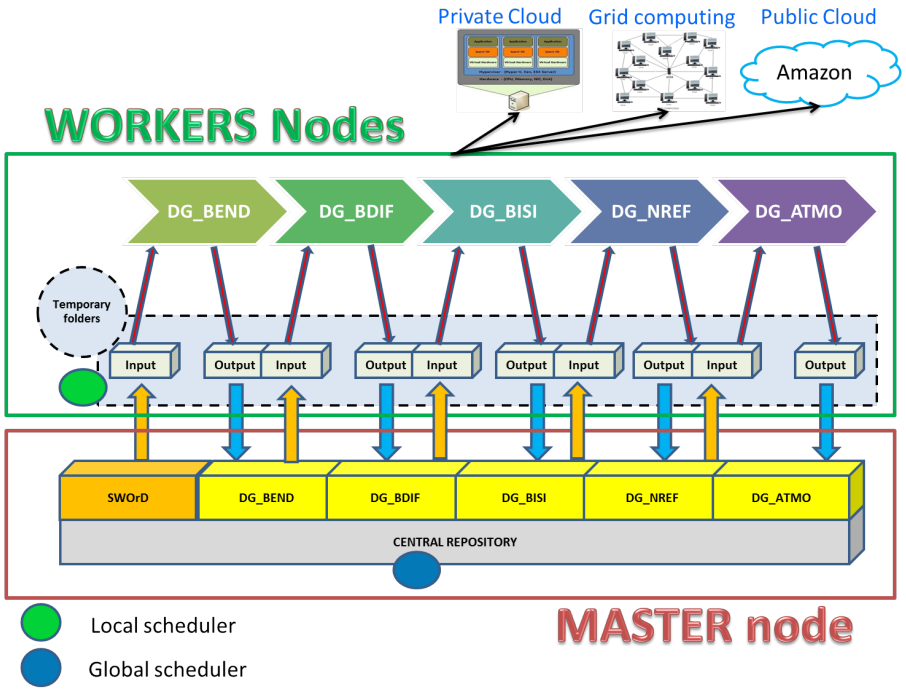

Figura 2.10. Grid Rosa folders

2.5.6 Scalability on cloud computing

The Hybrid Architecture [1, 2] consists of a virtualized nodes integrated and connected into a grid infrastructure composed of physical nodes. With physical node, it's mean a machine that runs an operating system that has the exclusive use of the underlying hardware. The virtualized node is instead an instance of a virtual machine that can share resources with other nodes (see Fig. 2.11), managed by the hypervisor for the Private Cloud and managed by Amazon for the public Cloud.

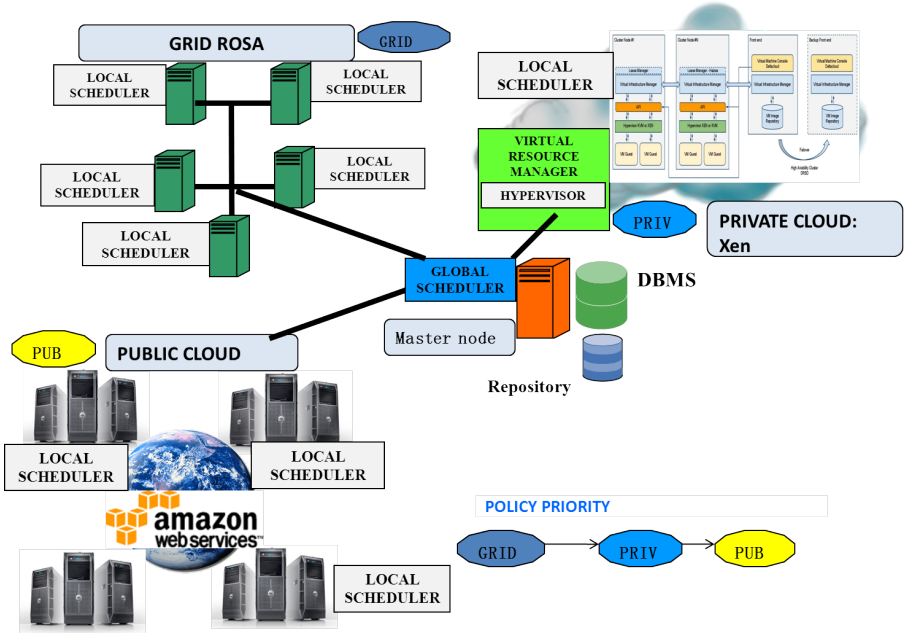

Figura 2.11. Hybrid Infrastructure

The project stems from need of computational power in case of an unexpected burst of calculation that the physical infrastructure would not be able to respond on its own. In these cases, where the physical grid has saturated its resources, the global scheduler provide to ask additional request to the hypervisor for new virtualized nodes which are according on rules sets in the scheduler. This architecture allows to profit from the grid (increasing computing power through the pooling of resources calculation, etc.) and from the virtualization (flexibility, scalability, cost reduction, etc.).

The use of virtual nodes in addition to the physical nodes in the grid has considerable advantages. Virtual machines can be blocked and quickly reboot every time

you need, without loss of information or problems to the chain flow of execution during the processes on the machine. Tests were performed on virtual machine to estimate startup time and the result is about 9500 ms. In addition to reduced start-up time, the use of VM in the grid brings other benefits including load balancing and high availability. The load balancing allows migration of virtual machine from a physical box to another, in order to balance system performances; an high available system ensures migration of virtual machine when maintenance shall be paid on the server, avoiding possible (and usually lengthly) discontinuity in service provisioning. Startup time is the difference in milliseconds from the time when hypervisor receives a request to start the VM to the time when the VM is accessible on the network, and ready for a job execution. On figure 2.12 an overview from web site developed concerning the status of the infrastructure.

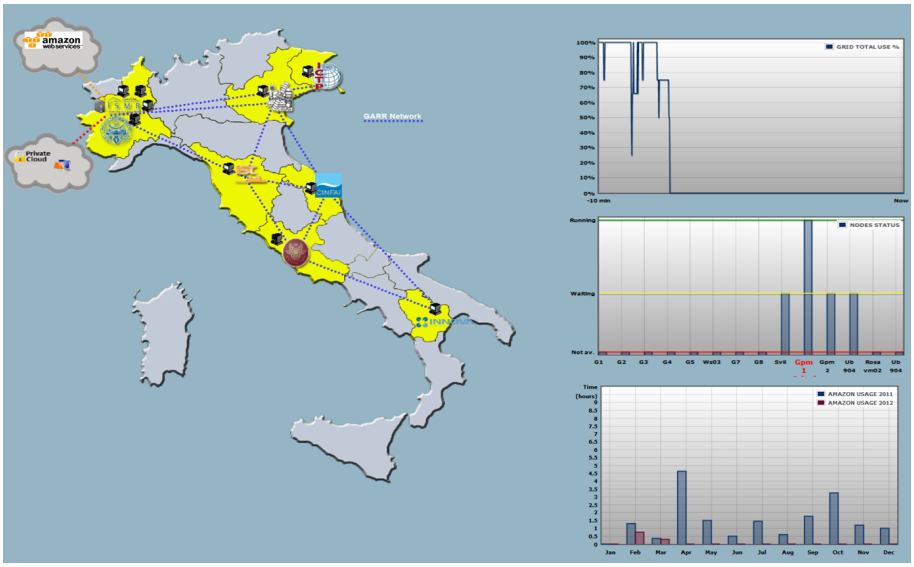

Figura 2.12. Web site: infrastructure overview

The Xen hypervisor is a layer of software that replaces the operating system running directly on the hardware of the computer. It is released under GPL license for x86-compatible and was developed at the University of Cambridge. It was decided to use Xen as virtualization platform because it is an open source product and is one of the few hypervisor that supports both paravirtualization and full virtualization. Xen Hypervisor is the direct interface between virtual machines and the hardware, and receives all requests for CPU, I/O and disk usage. Due to the separation between the OS and hardware, the hypervisor can run multiple operating systems safely and concurrently. The Domain 0 (Dom0) is the only domain that

can access to the hypervisor and can manage (create, shutdown, suspend, etc.) the virtual machines.

The Domain Guests (DomU) are controlled by the Dom0 and can operate independently in the system. The DomU can be of two types: paravirtualized or fully virtualized, also called Hardware Virtual Machine (HVM).The paravirtualized systems are aware to have no direct access to the hardware and they use special instructions to interact with the kernel. In this case, the virtual machine operating system must be adapted for the purpose. HVM machines are not aware that they are virtualized and interact with the hardware as if they were running directly on it: in reality all messages are filtered by the hypervisor. The xm comands are the main interface for managing the guest domain. Thanks to them, it is possible to create, pause, and shutdown the VMs, but also enable or pin VCPUs and attach and detach virtual block devices. The commands are listened by a daemon called xend.

RESOURCE SCHEDULER As mentioned above, in case of saturation of physical resources, the system is able to automatically start virtual machines: it can support the grid taking in charge the execution of jobs. For this purpose, it has been designed and implemented a Resource Scheduler that decides to allocate new virtual machines, depending on specific dynamics. The scheduler is a process running on the hypervisor and consists of several bash scripts allowing the monitoring and the collection of data which will be used for calculating system parameters and for the generation of log files [8, 9].

The logic model determines the allocation of the VM, it is based on the observation of the status of the grid: specifically processes execution and computational resources availability. The information are retrieved by querying the database hosted on the master node of the grid, in which data are collected periodically from each node, through a push mechanism. The resource scheduler starts a VM and sends to the master node information about availability of new node just started.

As soon as the master node, where resides the job scheduler, detects the new virtualized node, jobs are assigned on virtual machines on the same way and when all process are finished, the resource scheduler switch off the VM. The scheduler, once this limit is exceeded, tell to the master node that the virtualized node is no longer available to receive job. Then it will proceed to shutdown the machine, after to have verified that there are no file transfers in progress, there is not a queue of files to be processed, the processing job is considered completed.

During a test phase we evaluated the elaboration time of each Data Generators testing between two types of nodes: physical and virtualized node on both private

and public cloud. For private Cloud the server used for testing is equipped with a dual-core Intel Xeon (4 CPU), 8 GB of RAM and 130 GB of storage, for public cloud instances used were large instance with the same capabilities. The operating system is Ubuntu server.

The guest machines reside entirely on this server and therefore they share the resources (RAM, CPU, disk): each machine has 2 GB of RAM and 2 dedicated CPUs. Virtualized nodes are configured exactly like a physical node of the grid. It has been installed the softwares used for the chain processing and some system tools for the local job scheduling and monitoring of the resources. It was decided to use paravirtualized systems since it was shown that (in terms of network and I/O), they have better performances than the fully virtualized one [10]. The execution time of algorithms DG BDIF, DG BISI, DG NREF executed on the virtualized machine are almost comparable to the execution time on the physical machine. However, if the algorithm is executed on the virtualized machine DG ATMO has a slight delay, estimated in 5 %, compared with physical machines due to the time for starting up virtual machines.

2.5.7 Improving performances

The DGs for the overall processing chain were tested during a learning phase [12]. For a single event we obtained the percentage values and also for daily events. The amounts of input, output and time for elaboration were considered. We have assume that SWOrD has already been executed, and then it is outside the calculation. The execution time trend is estimated when the number of nodes and events is increased.

When only one node is available, the total execution time for daily files is 1752 minutes (about 29 hours). Instead of increasing the number of nodes, the execution decreases further, just note that with 2 nodes it is 912 (about 15 hours). This means that, when a single event is processed, there is no gain time in the grid environment. The time is higher rather because we must consider the transfer time. It has sizeable gain time only when a set of files are processed. Certainly, the benefit of the grid is ensuring the elaboration of the overall chain in less time.

In a distributed system where worker nodes are geographically located, it can have disadvantages in the network layer in the case of network failures or slow connections. To overcome this problem, only internal nodes are available for elaboration. We executed some tests for each DG algorithm to evaluate the performance of the system. The aim of the first test is to obtain an estimate of the total processing time for each DG. The procedure involves the execution of the six jobs, on each Grid node. The test was repeated several times in order to achieve an average

value for the execution time of each DG on each node.

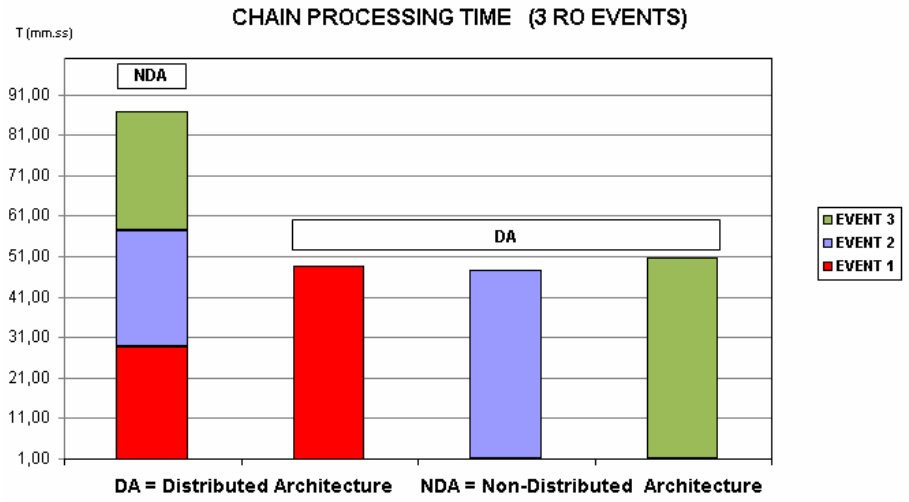

Figura 2.13. Execution time

In our case all the Grid nodes are 64 bits Dual Core 2 GHz with 2 of GB Ram, hence since all machines have the same characteristics, the performances are very similar. The second group of tests considers the execution of three distinct events in order to make a comparison between a non distributed and a distributed architecture. Thanks to this evaluation we can notice that, in general, in the DA the execution time of the three events is considerably shorter than the same execution time in the NDA. In Figure 2.13 we can observe that the total execution time in the DA reaches a peak at 51,10 minutes (the maximum time among the three executions) while in the NDA the total execution time is about 87,51 minutes (the sum of the three events). Actually, with an in-depth code optimization, SWORD processing time is one order magnitude lower.

Capitolo 3

Virtual Grid and Cloud Computing in Computational Electromagnetics

3.1 Introduction

Computational Electromagnetics (CEM) is scientific discipline aimed at simulating the interaction of electromagnetics fields with structures in a particular environment. The geometries analyzed are typically very complex and they could be simulated starting from a CAD model, possibly including fine levels of detail.

Many macroscopic phenomena in a great number of fields are governed by this set of differential equations: electronic, geophysics, medical and biomedical technologies, virtual EM prototyping, besides the traditional antenna and propagation applications. Therefore, many efforts are focussed on the development of new and more efficient approach to solve Maxwell's equation.

The interest in CEM applications is growing on. Several problems, hard to figure out few years ago, can now be easily addressed thanks to the reliability and flexibility of new technologies, together with the increased computational power.

This technology evolution opens the possibility to address large and complex tasks. Many of these applications aim to simulate the electromagnetic behavior, for example in terms of input impedance and radiation pattern in antenna problems, or Radar Cross Section for scattering applications. Instead, problems, which solution requires high accuracy, need to implement full wave analysis techniques, e.g., virtual prototyping context, where the objective is to obtain reliable simulations in order to minimize measurement number, and as consequence their cost. Besides, other tasks require the analysis of complete structures (that include an high number of details)

by directly simulating a CAD Model. This approach allows to relieve researcher of the burden of removing useless details, while maintaining the original complexity and taking into account all details. Unfortunately, this reduction implies:

a high computational effort, due to the increased number of degrees of freedom,

b worsening of spectral properties of the linear system during complex analysis.

The above considerations underline the needs to identify appropriate information technologies that ease solution achievement and fasten required elaborations. Cloud Computing technology can be very useful to these purposes thanks to the capabilities to elaborate enormous amounts of data and and to for capabilities to enable dynamic large-scale resource to solve problem by exploiting distributed scenarios. The main advantage of using a distributed infrastructure is due to parallel computing, indeed if a problem can be split in smaller tasks, that can be executed independently, its solution calculation fasten up considerably. The distributed infrastructure thanks to virtualization technics can be easily setup by using cloud computing technology for resources management [16].

To exploit the advantage of virtualprototype, it is necessary to identify a technique able to split original electromagnetic task into a set of smaller subproblems. This is reflected in a high computational effort that must be faced by high performances infrastructures. On this sense the use of cloud technology addresses the potential benefit of adopting a new way to solve CEM problems. In particular the focus is on a Domain Decomposition (DD) technique to reduce the complexity of a MoM analysis of large and complex structures. The Domain Decomposition (DD) based on the block generation algorithm introduced in Matekovits et al. perfectly addresses the requirements for adapting applications analysis in a cloud context technology. The main idea is on integration between CEM and modern computing infrastructure for reducing time processing on CEM analysis and for giving a new approach in term of computational aspect in virtualprototype context.

3.2 Domain Decomposition method for cloud: Motivation

An integration of CEM problems in a distributed infrastructure introduce a significant change for the way who CEM applications and algorithms are implemented if we considers the availability of more than one resources for process analysis [15]. Due to the decomposition of an original problems into a larger number of subproblems, the scheme is well suited for a parallelization analysis approach, since the subproblems be referred to as blocks are disjoint, and the elaboration of the blocks is intrinsically a parallelization operation. Since the generation of the entire domain basis functions on each block is independent from the other blocks, a high scalability is expected.

As an example, the figure 3.1 shows a sphere subdivided into 8 different blocks, identified by different colors. The eight blocks can be processed in parallel by four different processes, in order to generate the basis functions describing each block.

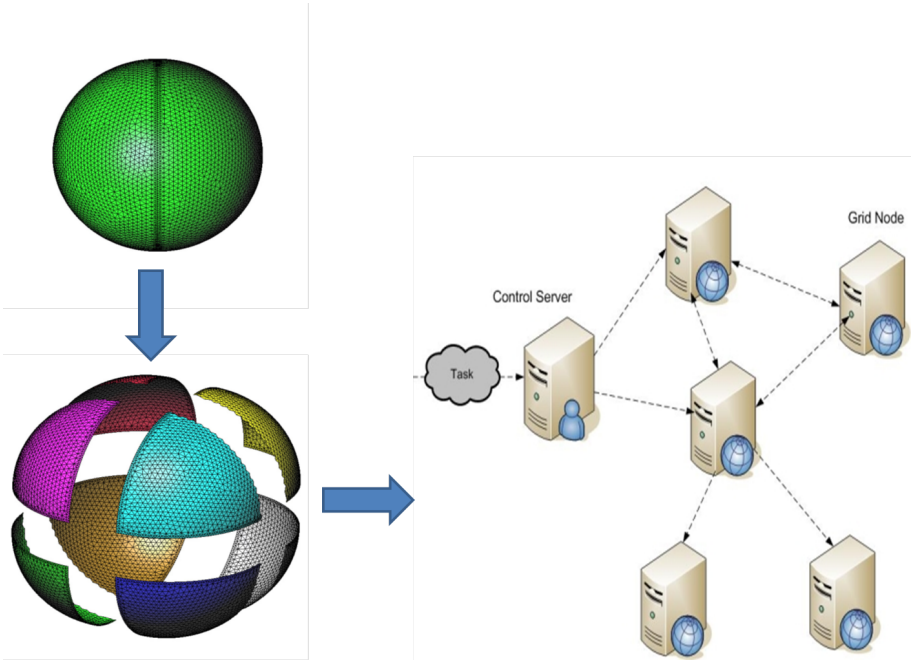

Figura 3.1. Decomposition of a sphere

The new parallelization approach can be achieved using two different techniques: parallel programming or distributed processing.

For instance Parallel programming is the technique by which have been obtained the best results in this field (500 billion of unknowns) Mourio et al. (2009). There

are important barriers to the broader adoption of these methodologies though: first, the algorithms must be modified using parallel programming API like MPI (2011) and OpenMP (2011), in order to run jobs on selected cores. The second aspect to consider is the hardware: to obtain results of interest supercomputers with thousands of cores and thousands of GB of RAM are required. Typically these machines are made by institutions to meet specific experiments and do not always allow public access.

The Grid Computing and Cloud technology is a rather more easily applicable model for those who do not fulfill the requirements listed above Foster and Kesselman (2003). For its adoption the only requirement is to have at disposal computers or server for virtualization to use. The machines may be heterogeneous, both in terms of the types (workstations, servers) and hardware resources of each machine (RAM, CPU, HD).

With the Virtualization technology:

A first real advantage is the number of processes running. Each machine (for multicore ones) can be optimized by instantiating multiple virtual nodes on the same physical machine. In a single powerful server it's possible to set more than one machine with the main advantage who base of the needs and requirements of jobs to be analyzed the number of CPU and dimensions of RAM memory can be set and pre-configured. In fact Virtualization is a enable technology for having a flexible way to manage a powerful hardware it's common to have several machine running with different configuration in term of computational capabilities; for example two virtual machines with 8 CPUs and each with 32Gb RAMs and 2 virtual machines with 4 CPUs and 16Gb RAM memory. This is a significant change if we consideres that in a traditional use of servers only one machine was active by using the full power capacities of the hardware. Is frequent that in most cases for process analysis only few CPUs and RAM memories will be used not the full power. Virtualization is an help for improving and optimizing the use of hardware resources. It's a way to use exactly what we need, in an on-demand scenario.

The second real advantage of this model, however, is that researchers do not have to worry about rewriting code to make their application parallelization. The concept is to move a physical machine to a virtual machine by maintaining operating systems and applications installation and configurations. In this way it's possible to pre-configure and specialize machine for specific analysis and to choice the right pre-configured systems at the right time. A common scenarios is to have a list of images (Operating System and applications) a server available, typically a user after the right choice of pre-configured image, CPU and RAM capabilities instantiate the

virtual machine for process analysis, at the end of process analysis he release the virtual machine. In this way the server or a pieces of server capability is free of new instantiation. The approach is a multi-users context with virtual machines available on the same server by running or shutting down instance.

One of the drawbacks in the adoption of this model is the addition of overhead introduced by the distributed infrastructure (e.g., inputs time transfer) and by the hypervisor that manages the virtual machines. It has been shown, however, that the loss of performance due to the hypervisor for virtualization is not particularly high, usually not exceeding 5 % Chierici and Verald (2010). For the above reasons, the use of virtual grid infrastructure is adapted by using, heterogeneous machines, both physical and virtual, on which to run scientific applications without the need to modify the source codes of the used algorithms.

In order to simplify the mathematical model, the electromagnetic behavior of Perfectly Electric Conducting (PEC) objects in a free space environment will be briefly introduced. The described approach is not limited to PEC objects in free space, in fact it is applicable to different formulations as well (dielectric objects or layered media problems for instance): in other terms it is a kernel free method. Besides, the focus is on the use of distributed architecture computing approach applied to computationally demanding electromagnetic problems: rather than considering more complicate approaches, that would divert the attention from the subject of work. The choice for the activities was to prefer to introduce and apply the method to PEC objects in a homogeneous background, but it can be applied to other formulations as well. The Electric Field Integral Equation (EFIE) is a very versatile approach to the full-wave analysis of complex electromagnetic problems: for PEC objects the EFIE can be written by enforcing the boundary condition on the surface of the object, i.e.

The tangential component of the electric field vanishes on the surface S:

$$\hat{n} \times \underline{E}|_{\Sigma} = \hat{n} \times (\underline{E}scat + \underline{E}inc)\,|_{\Sigma} = 0 \tag{3.1}$$

The surface S is discretized by a mesh with triangular cells, over which a usual system of RWG functions $\underline{f}_n$ is defined. The unknown surface current $\underline{J}$ is approximated by the above set of RWG basis functions

$$\underline{J}\,(\underline{r}) \approx \sum_{n=1}^{N} I_n \underline{f}_n\,(\underline{r}) \tag{3.2}$$

A Galerkin testing is used to convert the EFIE into the MoM linear system; hence we obtain the matrix equation

$$[Z] \cdot [I] = \left(\left[Z^{\phi}\right]\left[Z^{A}\right]\right) \cdot [I] = [V] \tag{3.3}$$

where a generic element of the scalar potential and of the vector potential matrix, $\left[Z^{\phi}\right]$ and $\left[Z^{A}\right]$ respectively, is expressed as

$$Z^{\phi}_{m,n} = \frac{1}{4\pi j\omega\epsilon_0} \iint_{S_m} dS\ \nabla_s \cdot \underline{f}_m(\underline{r}) \iint_{S_n} dS'\ G(\underline{r},\underline{r}')\ \nabla_s \cdot \underline{f}_n(\underline{r}') \tag{3.4}$$

$$Z^{A}_{m,n} = \frac{j\omega\mu_0}{4\pi} \iint_{S_m} dS\ \underline{f}_m(\underline{r}) \cdot \iint_{S_n} dS'\ G(\underline{r},\underline{r}')\ \underline{f}_n(\underline{r}') \tag{3.5}$$

where: $G(\underline{r},\underline{r}') = \frac{e^{-jk_0R}}{R}$, $k_0 = \omega\sqrt{\epsilon_0\mu_0}$, $R = |\underline{r} - \underline{r}'|$, and S_m is the definition domain of the function $\underline{f}_m$. The coefficients I_n in (3.2) are collected in the vector $[I]$, and the mth element of $[V]$ is equal to

$$V_m = -\iint_{S_m} dS\ \underline{f}_m(\underline{r}) \cdot \underline{E}^i(\underline{r}) \tag{3.6}$$

where $\underline{E}^i$ is the impressed field in absence of bodies.

The first step of the Domain Decomposition approach is a subdivision of the overall geometry, breaking down the scatterer surface S into NB sub-scatterers, which will be referred as blocks in the following. An example of the splitting of a plane into blocks is shown in Figure 3.2.

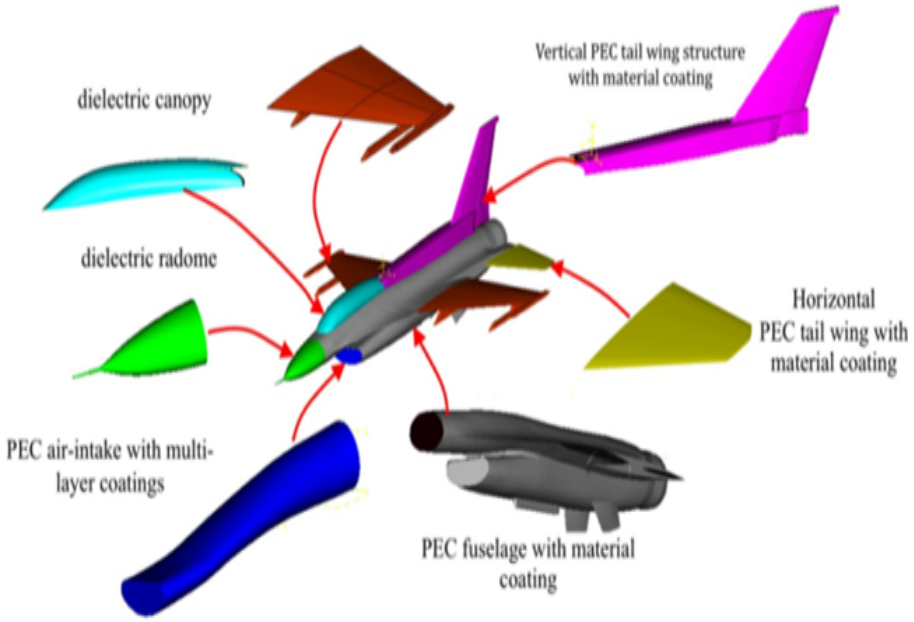

Figura 3.2. Decomposition of a plane

To subdivide the structure in blocks, on which entire domain synthetic functions will be generated, a fully automatic procedure was used. For the implementation

and adaptation in a cloud distributed architecture this aspect it is of vital importance to be able to properly generate the subdomains for the synthetic function approach, and it is especially critical for arbitrary and complex structures.

The block generation algorithm is based on the multi-level cell grouping described in Andriulli et al. (2008); Vipiana et al. (2009) for the generation of the Multi-Resolution basis functions. The starting point is the usual mesh for the analysis of the considered structure, without any constraint on the mesh properties and on the topology of the structure. Then a nested family of meshes, with non-simplex cells, is generated through subsequent groupings of the initial triangular mesh cells. We denote this initial mesh with M0, and call it level-0 mesh. All other meshes will be composed of groups of adjacent cells of the initial mesh. the starting point is to considering groups of adjacent cells in M0 formed so that their average area is about four times the average area of the cells in M0. This covering will be called the level-1 mesh, M1. The same procedure applied to M1 will generate the generalized mesh M2, and so forth. Grouping a cell was stop with its adjacent cells when its size reaches a chosen threshold.

The algorithm stops when the last level L contains non-simplex cells with (linear) dimension around G. The non overlapping blocks, on which the synthetic functions will be generated, will then be the cells of the last level ML. The only parameter the user has to control for the algorithm is the threshold where the grouping has to be stopped. We underline that the grouping procedure used to find the domains where the Synthetic Functions are defined has a O(NlogN) complexity, where N is the number of initial mesh cells.

The next step consists in the generation of the basis functions to model the current distribution over each block; these will be referred as synthetic functions (SFs), whose support extends over the entire block. The synthetic functions are chosen in the set of responses to the required incident field, and to other sources placed at the borders of the block and around it, to make up the space of all (rigorous) solutions restricted to that block. Once the set of the solutions to all these sources is computed, the minimum number of necessary responses has to be determined. This is done through a combination of a Singular Value Decomposition (SVD) and a Gram-Schmidt (GS) procedure, applied to the matrix that collects the responses from all the sources: the functions are then selected with a proper thresholding on the associated singular value.

Finally, the set of RWG functions defined over the connections of contacting blocks is added to the set of SFs in order to guarantee continuity across blocks. Since the SFs are expressed as a linear combination of the initial RWG functions,

the scheme can be seen as a purely multiplicative algebraic compression of the standard MoM matrix. It should be clear now that the process of generation of the synthetic functions is carried out independently on different blocks, i.e. the set of SFs is generated by solving the electromagnetic problem on each block in isolation.

As a consequence, a distributed architecture approach is particularly well suited for the generation of the SFs: each block can be processed by a different node of the distributed infrastructure. Finally, after the complete set of SFs is generated, the original system matrix is compressed, and the compressed system is inverted to yield the solution of the full problem. The goal of the synthetic functions approach is to accelerate the solution of the problem when a large number of excitation vectors (right hand sides, RHSs) is present, which is typical Grid Infrastructure for Domain Decomposition Methods in Computational ElectroMagnetics.

In order to exploit the distributed infrastructure approach, each node of the virtual grid has to sequentially solve a number of blocks [17]. The synthetic functions are chosen in the set of responses to the required incident field, and to other sources placed at the borders of the block and around it, to make up the space of all (rigorous) solutions restricted to that block. Since the grid is in general heterogeneous, i.e., the nodes have different processing and memory characteristics, the blocks should be properly dimensioned and assigned to the computing nodes. When the block dimensions are properly taken into account, the computation of the full MoM matrix and its handling do not pose a limitation within a single block. However, once the full set of synthetic functions is generated, one needs to apply the basis change to the full system matrix related to the original structure, in order to compress the system and invert it. For practical problems this is not feasible though, due to the total dimension of the problem. Therefore the compression was performed within a fast scheme, which avoids computing the full system matrix and makes the solution of large problems possible. The compressed matrix in the new basis can be written as:

$$[Z_{SF}] = [T]\,[Z]\,[T]^H \tag{3.7}$$

where $[Z_{SF}]$ is the compressed matrix, $[T]$ is the change of basis matrix, whose dimensions are $N_{SF} \times N_{RWG}$ (the total number of synthetic functions and the total number of RWG functions, respectively), $[Z]$ is the MoM matrix in the RWG basis, and $[]^H$ represents the hermitian operator. The same change of basis is performed on the RHS, namely:

$$[V_{SF}] = [T]\,[V] \tag{3.8}$$

where $[V_{SF}]$ is the compressed RHS in the SF basis. Finally the compressed system is solved. At the present stage of the work, the compression and the solution of the

complete system is not carried out in the grid though; this will be object of future research.

3.3 Virtual grid computing

Due to the decomposition of the original problems into a larger number of subproblems, the scheme is well suited to a parallelization approach, since the subproblems (which will be referred to as blocks in the following) are disjoint, and the elaboration of the blocks is intrinsically a parallelizable operation. Since the generation of the entire domain basis functions on each block is independent from the other blocks, a high scalability is expected.

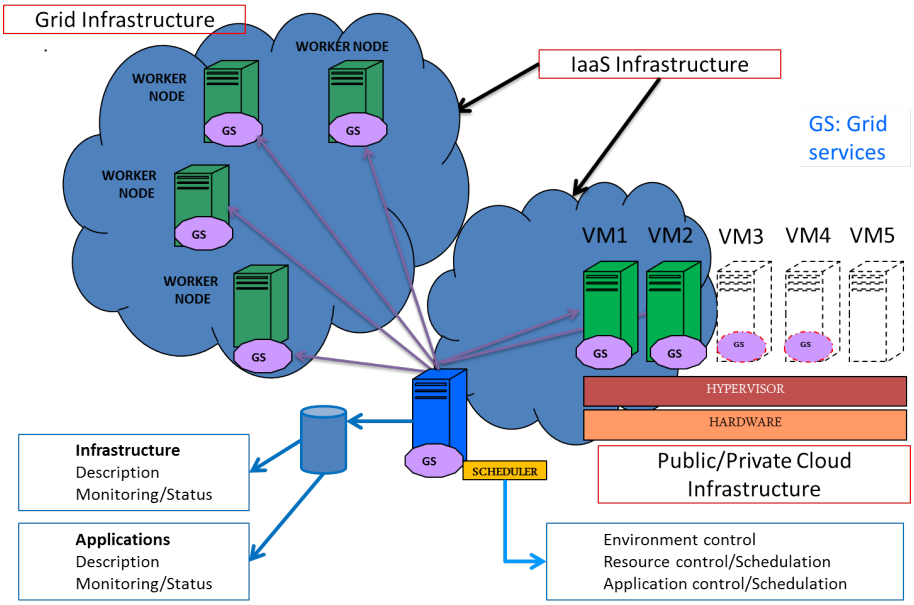

Figura 3.3. Jet fighter test case

Due to the rapid technology growth in the recent years has helped the development and rapid expansion of Cloud Computing technology. In the late 1990s began to emerge, a more generalized framework for accessing high performance computing systems, and at the turn of the millennium, the pace of change was accelerated by the recognition of potential synergies between the grid and Cloud computing thanks to the virtualization.

Virtual grid extend the concept of grid by using virtual machine see Fig 3.3), Grid computing paradigm is conserved but not only with physical machine. For that it was maintained grid services by using the emerging Service Oriented Architectures

(SOA) through the creation of the Open Grid Services Architecture (OGSA) Kourpas (2006). The Open Grid Services Architecture is a set of standards defining the way in which information is shared among several components of large, heterogeneous grid systems.

The OGSA is, in effect, an extension and refinement of the Service Oriented Architecture OGSA (2007). The Open Grid Services Architecture is a standard created by the Open Grid Forum (OGF) OGF (2011). OGF was founded in 2006 from the Global Grid Forum (GGF) and the Enterprise Grid Alliance (EGA). GGF had a rich background and established international presence within the academic and research communities while EGA was focused on developing and promoting enterprise grid solutions. The OGF community now counts thousands of members working in research and industry, representing more than 400 organizations in 50 countries.

Grid Computing is often described by referring to the analogy between electrical networks and grid. When people access to electric network they use wall sockets with no care about where or how electricity is actually generated. This relation underlies that computing becomes pervasive thanks to Grid Computing diffusion. Therefore, individual users (or client applications) can access computing resources (processors, storage, data, applications, etc..) when needed with little or no knowledge about where those resources are located or what underlying technologies, hardware, operating system are used.

A further definition is given by Grid is an infrastructure that involves the integrated and collaborative use of Grid Infrastructure for computers, networks, databases and scientific instruments owned and managed by multiple organizations Asadzadeh et al. (2005). Grid Computing is based on these technology principles Oracle (2009): Standardization on operating systems, servers, storage hardware, middleware components, and network components extends interoperability and reduce system management overhead. It also improves operational complexity reduction in data center by simplifying application deployment, configuration, and integration.

The adding step for an evolution from Grid to Virtual gird pass by Virtualization of resources means that applications are not bound to a specific server, storage, or network components but can be used in any virtualized resource. Virtualization is realized thanks to a sophisticated software layer that hides the underlying complexity of hardware resources and presents a simplified interface used by applications and other resources.

Automation Grid Computing requires large-scale automation of IT operations due to the potentially very high number of components, both virtual and physical.

Each component requires configuration management, on-demand provisioning, monitoring, and other management tasks. Combining these capabilities into a single, automated, integrated solution for managing grids enables organizations to maximize their return of investment.

For an easy use of both resources physical and virtualised, the Globus Toolkit (Globus (2010)) developed by the Globus Alliance was used for services. Globus is an open source software toolkit used for building grid systems and applications. The Globus Alliance includes ISI, the University of Chicago, the University of Edinburgh, the Royal Institute of Technology in Sweden, the National Center for Supercomputing Applications, and Univa Corporation. Sponsors include federal agencies such as DOE, NSF, DARPA, and NASA, along with commercial partners such as IBM and Microsoft. The Globus Toolkit includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability. It is packaged as a set of components that can be used either independently or together to develop applications. It is used by various companies and organizations as the basis for grid implementations of various types. Globus Toolkit is composed by four main components (see Fig. 3.4):

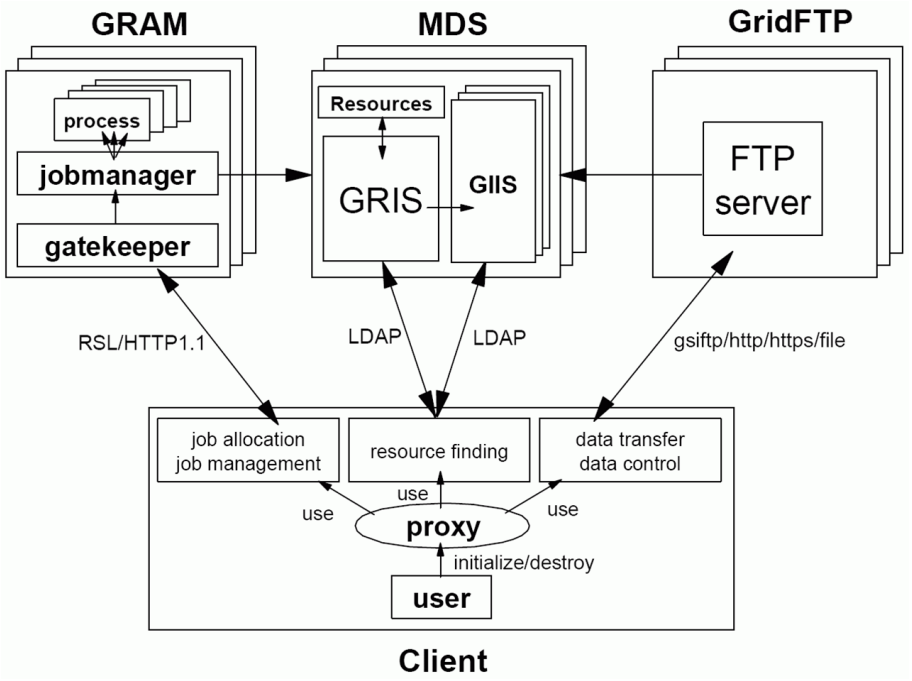

Figura 3.4. Globus:Main components

- Security (GSI: Grid Security Infrastructure). It is a set of tools, libraries and protocols used in Globus to allow users and applications to securely access resources. GSI is based on a public key infrastructure, (PKI) with certificate authorities (CA) and (X509) certificates. GSI uses (SSL) for authentication and message protection and enables user to create and delegate proxy credentials to processes running on remote resources;
- Resource Management (GRAM: Grid Resource Allocation Manager). It provides the user to access the grid in order to run, terminate and monitor jobs remotely;
- Information Services are for providing configuration and adaptation capabilities for heterogeneous resources: 1)MDS: Monitoring and Discovery Service: provides information about the available resources on the Grid and their status; 2)GRIS: Grid Resource Information Service: is associated with each resource and answers queries from client for their current configuration, capabilities and status; 3)GIIS: Grid Index Information Service: is a directory service that pulls information for GRIS's. It is a caching service which provides indexing and searching functions.
- Data Management GridFTP: is a protocol that provides for the secure, robust, fast and efficient transfer of data.

The basis idea of pooling has been always employed by humans. Its most evident and valuable advantage is cost and resource optimization, but it hides facets that may shadows its benefits. Whenever we share something we are worried since our goods may not be handle properly and may be manipulated by strangers. Moreover when we use someone else stuff we worry about safety since objects may be dangerous, broken or compromised. The above concept perfectly fits grid system since grid can be seen as a mechanism to pool resources to optimize system utilization. Therefore authentication and authorization of users, and data protection, are essential.

3.4 Cloud4CEM infrastructure

The Cloud4CEM activity aims to improve performance of CEM application. This objective is achieved by the adoption of a Virtual grid architecture that exploits the benefits derived from the parallel computation [18]. The proposed system manages data automatically and without impact for users. Furthermore, virtualized resources make system flexible, simplify the underlying infrastructure and improve scalability. To this aim an hybrid grid cloud infrastructure was built and tests were performed in order to compare grid results to the ones of the same task executed on a single machine.

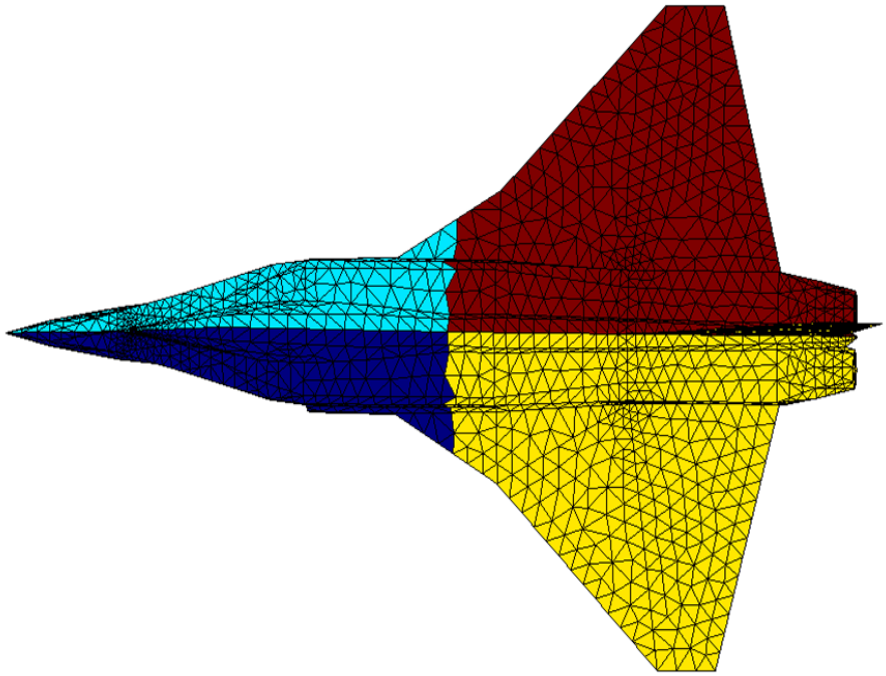

Figura 3.5. Jet fighter test case

The virtual grid is composed of two types of nodes, a Master Node (MN) and Worker Nodes (WN). The Master Node is in charge to manage, control, and grant the security of the entire infrastructure. In addition, it coordinates other node operations. The middleware used to create the grid infrastructure is the Globus Toolkit (Globus (2010)). The toolkit, as explained in previous section, includes functionalities for security, information systems, resource and data management, node communication, fault detection, portability and all the features need to safely

deploy grid.

The input data is a file that contains the mesh structure to analysis. It is split by the MN in smaller files, called blocks, that are distributed to WN. Since these blocks are generated according to Domain Decomposition technique and are independent, they can executed in parallel. The size of split blocks is not uniform but depends on the geometry of the structure. Moreover, the MN hosts the Global Scheduler (GS), that represents the smart component of the grid, and it holds information related to grid status, that are sent from each Worker Node, and stored in a database. The WN, instead, receives blocks, elaborates them and sends results to MN.

Virtual Grid computing is a distributed computing infrastructure, consisting of heterogeneous computing resources, used to process and manage a large quantity of data. Essentially, in virtual grid computing, a single big task is split into multiple smaller tasks which are further distributed to different computing machines. Upon completion of these smaller tasks, they are sent back to the primary machine which reassembles the partial outputs yielding a single output. The crucial aspect that makes grid computing different from the Message Passing Interface (MPI) parallel technique is that MPI is essentially a programming paradigm that allows for taking extremely large sets of data and crunching the information in parallel while sharing the data between computing nodes. Certain kinds of problems necessitate this sharing as the computed results on ne node may affect the computed results on another node in the grid. MPI-based clusters are therefore designed and built in order to maximize throughput and minimize the latency.

This is not the case for Virtual Grid architecture: smaller tasks, independent from each other, are executed by different computing nodes. At last, the primary machine, called Master of the infrastructure, reassembles the partial results obtained from the computing nodes, yielding a single output. Another important aspect to keep in mind is the non-necessity to heavily rewrite source code to make it parallel (as in the case of MPI parallelization).

Due to the decomposition of the original problem into a large number of sub problems, generation of the entire domain basis functions on each block is independent from the other blocks indeed. As an example, the grid approach is applied to the SFX technique.

Virtual Grid computing is a rather more easily applicable model for those who do not fulfill the requirements listed above. For its adoption the only requirement is to have at disposal computers to use within the grid. The machines may be heterogeneous, both in terms of the types (workstations, servers) and in terms of hardware

resources of each machine (RAM, CPU, HD); besides they can also be recovery machines. With the Virtualization technology the number of processes running on each machine (for multicore ones) can be optimized by instantiating multiple virtual nodes on the same physical machine. Virtualization enable horizontal and vertical scalability. Horizontal by adding or deleting nodes based on analysis request. Vertical by changing or setting cpu number and RAM size base of jobs requirements in term of computing capabilities.

For the distributed infrastructure implementation it was decided to build a low-cost computing infrastructure, composed of heterogeneous machines, both **physical** and **virtual**, on which to run scientific applications without the need to modify the source codes of the used algorithms. Main goal is the reduction of execution times of CEM applications. In order to test the effectiveness of the grid, test-case is a jet fighter aircraft (see Fig. 3.5) has been discretized with a linear mesh density around 5cm, corresponding to 156K input file 17Mb. The plane is illuminated with a wave at the frequency of 600 MHz. The geometry has been subdivided into 67 blocks, each one described in a mesh file of about 7.5 MB in order to apply the SFX approach.

A pictorial representation Cloud4CEM architecture is shown in Figure 3.6: a Master node is in charge to manage, control, and grant the security of the entire infrastructure. In addition, it hosts the Global Scheduler (GS), responsible for the coordination of other node operations: it processes the geometry to be analyzed, and splits the job into smaller tasks to be assigned to the available worker nodes (WN), balancing he computational burden of each WN. Each WN is provided with a Local Scheduler (LS), in charge of starting jobs as requested by the MN, tracking the status of a job, and sending back the computed results to the MN.

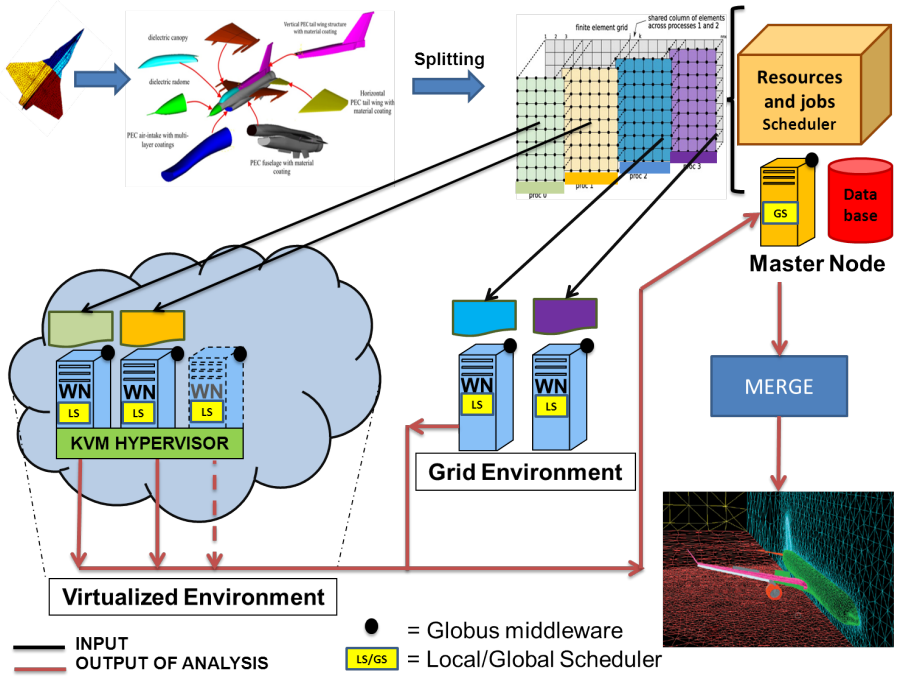

Figura 3.6. CloudCEM Infrastructure

The execution phase on each node is managed by a Local Scheduler (LS), the aim of LS is to check is the node is on RUNNING or WAITING status and to run jobs. General status of a node status is monitored by an always active agent, both LS and Agent are installed on each worker node. The agent monitors is is charge to check the availability of each service on the worker node and sends periodically its status to the database on the Master node (MN). The Monitoring Agent (MA) role is crucial for the global scheduler since the choice to send a job to a node depends on the information provide by MA. If all services are available and the node is in WAITING condition for processing jobs, the node is in condition to receive a job. All Worker Nodes have same operating system and are equipped with the same software: the middleware, the monitoring agents, the Local Scheduler and the

applications code for execution. The Master Node is in charge to manage the global virtual grid infrastructure that is the responsibility of the Global Scheduler. It is a software module developed in Java.
Two main features are assigned for the GS:

The **first** concern the distribution of smaller parts of the input data (blocks) to each Worker Nodes, for this it communicates with other nodes thanks to grid services provided by the Globus Toolkit. The files are sent by the scheduler in order to balance the execution on each node: this is achieved by checking WNs availability and the number of files to sent. In order to know the overall status of the grid, the GS queries the database and transfers file only to nodes that have communicate their status within a specific time period. It is worth noting that this monitoring system that periodically push data into the database instead of gather information from each machine, allows to reduce time and computational wastes Gradwell (2003).

The **second** main function for the global scheduler is to provide scalability of the infrastructure. By the knowledge of status of each nodes and status of all running jobs he is in condition to make an evaluation of the availability of nodes and to provide to wake up additional virtual nodes on the private cloud infrastructure. The principle is base on the status of each node, WAITING,RUNNING processes or NOT AVAILABLE. In this way the global scheduler know how jobs are in queue and how nodes are free for jobs processing. In a scenario who no nodes are available or free for new jobs, GS provide to request additional Worker nodes by updating a table on central database.

On the same way on the private cloud computing a module make periodically a query for check new request of additional worker nodes and provide to to create a new instance. When a new instance is creates the MA installed update the central DB for a refresh of a new Worker nodes available and ready for processing jobs and consequently GS as explained before can send now jobs. This concept introduce the dynamic scalability in up and dow scaling. In fact when additional nodes end the job processing the instance is shutting down by the module installed on the hypervisor.

A job when is started update his status and the status of the node in order to know if a job is RUNNING, COMPLETED or FAILED.

Each WN is provided with a Local Scheduler, developed in Java, that checks contents of its input folder: every time a new job file is detected it executes the task. In order to be recognized the filename must follow a predefined naming convention rules. During the execution the analyzed block is transformed into a Method of Moments (MOM) matrix. If the execution terminates successfully, the output is sent to the Master Node, that reassembles it with the outputs received from other

nodes. Also the LS is in charge of tracking the status of job execution and sends information about start and stop time of each process.

The code in execution was not developed for parallel execution for this reason it was decided to optimize resources by virtualizing nodes in order to run multiple processes on the same physical machine. In this way it was possible to create multiple virtual nodes on the same resource and increase the available nodes number, e.g., the parallelization of the system, instead of coding parallelization, to improve the overall performance. Virtualized systems also help to improve infrastructure management, allowing the use of virtual node template to create virtual nodes in a short time, speeding up the integration of new nodes on the grid and, therefore, improving the reactivity and the scalability of the infrastructure. Another advantage of virtual environment is the quick recovery in case of damage of a virtual node. the virtual node blocked is shutting down and the system provide to choice the same template for a new instance, reducing de facto the downtime due.

The open source KVM (Hirt (2010)), has been used as hypervisor. It allows to create fully virtualized machines. The kernel component of KVM is included in mainline Linux. The basic requirements for the installation of this hypervisor is that the processor of the machine supports virtualization technology (Intel VT or AMD-V).

In order to measure the gain in terms of time, some performances test were conducted. It has been compared the result times of the analysis of a model executed on the grid infrastructure with the sequential execution of the same model on a single computer. The experiment allowed verifying the practical usefulness of the adoption of distributed infrastructures in this kind of applications. Nodes in use for the experimentation was:

Node	Type	CPU model	Virtual CPU	RAM[GB]
master	physical	Intel Core Duo 2.66GHz		4
wn1	physical	Intel Pentium 4 @ 3.20GHz		3.5
wn2	physical	Intel Core 2 6420 @ 2.13GHz		2
wn3	virtualized	Intel Xeon E5440 @ 2.83GHz	2	4
wn4	virtualized	Intel Xeon E5440 @ 2.83GHz	2	4
wn5	virtualized	Intel Xeon X3470 @ 2.93GHz	2	4

Tabella 3.1. Nodes Specifications.

The grid used for performance testing consists of hardware not purchased specifically for this purpose, but of computer available to researchers of Istituto Superiore

Mario Boella(ISMB). For this reason, its composition is heterogeneous in terms of the machines types (workstations, servers) and in terms of view of hardware resources of each machine (RAM, CPU, HD). Total execution time comparison between grid environment and sequential execution these machines three VMs have been created. To run the tests six nodes with the following configuration have been used: a Master Node, two physical Worker Nodes (wn1, wn2) and three virtualized Worker Nodes (wn3, wn4, wn5).

The first test performed was the execution of the entire process on a single physical machine. The node chosen for this test was the wn2: despite the smaller amount of RAM, the processing time of this machine is comparable to the average processing time of the other nodes.

The main input into smaller chunks and ran the sequential execution of individual blocks: the total processing time for 67 blocks on wn2 was equal to 6h 56min 51s. In the second test the splitting was delegated to the Master Node, which has also been responsible for the distribution of the single blocks to different nodes. The execution time of the build process of the blocks and the file transfer is 2min 55s, and therefore it is negligible if compared to the total execution time. The execution times of different nodes are very similar, only the virtual machine wn5 has slightly better performance, probably due to the higher performance processor. The total execution time of the grid is equal to the maximum execution time of individual nodes, i.e., 1h 42min 19s of wn1. The total time reduction is of 75 %.

For reasoning on the overhead introduced both by the grid and by the virtual machines. From the comparison between the execution of a group of blocks on the grid on a given node and sequentially on the same machine it can be deduced that the grid introduces an overhead (equal to 2min 38s in this case) due to the transfer of files output, but negligible compared to the total execution time.

Thanks to the Virtualization it has been possible to create multiple virtual nodes on the same physical machine, optimizing the hardware resources of nodes. In this way it was possible to increase the available nodes of the grid, i.e. the parallelization of the system, improving the overall performance. The virtualized systems also help to improve infrastructure management, allowing to use a template of a virtual node (previously configured with all the necessary software) to create virtual nodes in a short time, speeding up the integration of new nodes on the grid and therefore improving the scalability of the infrastructure. For the same reasons, the availability of nodes is improved since in case of damage of a virtual node the system will be able to quickly restore it, reducing the downtime due to his recovery.

As hypervisor the open source KVM has been used, which allows creating fully-virtualized virtual machines (VM), running an unmodified kernel. The basic requirement for the installation of this hypervisor is that the processor of the machine has virtualization support (Intel VT or AMD-V). The kernel component of KVM is included in mainline Linux. The master node that manages the security (it is the Certification Authority) is responsible for the scheduling of jobs and resources, and performs the splitting of the input data.

3.5 Cloud Computing for CEM: considerations

More and more applications require high performance computing, flexibility and reduced processing time. The architecture explained before, can be useful in many fields i.e., in e-Science applications (electronic, geophysics, biomedical domains).

The solution for CEM problems requires infrastructures based on high performance computing in order to reduce the execution time. However, it is not always possible to use supercomputers or parallelize algorithms code used for the calculation, a good solution may be represented by the Domain Decomposition technique, which allows the subdivision of the original complex problem into a set of smaller subproblems in a grid environment. For this activity, an Hybrid infrastructure was carried out, that consists of 6 nodes (both physical and virtual). The results showed that the adoption of this infrastructure has reduced the execution time with respect to the sequential execution on a single machine. It was also noted that the overhead introduced by the grid is negligible when compared to the total execution time.

The Virtualization has allowed optimizing the hardware resources of the machines, letting to run multiple blocks in parallel on the same physical machine, not introducing a significant overhead compared to the overall system performance. Studies are underway to improve the implementation of the scheduler, ensuring that blocks will be distributed to nodes most suitable for their execution. In particular, it will be developed a system that takes into account the weights of the jobs and the weights of the nodes available, assigned on the basis of predetermined criteria (e.g., file size, hardware resources, nodes availability, the average time of execution).

The Scheduler will also be able to turn on and off virtual nodes according to the needs of the system, considering the load level of the grid. Furthermore the virtual nodes will be configured dynamically as needed to perform specific jobs (RAM, CPU and storage). A further improvement will be the integration of the system with Public Cloud Platforms (e.g., Amazon EC2) in order to increase the system scalability, asking for virtual nodes usage only when necessary.

The high computational power needed to solve CEM problems requires the use of high performance computing infrastructures in order to reduce the simulation time. However, it is not always possible to use supercomputers and parallelize the code of the algorithms used for the calculation, and a good solution can be represented by grid infrastructures. For this project a grid of 6 nodes (both physical and virtual) has been realized, consisting of computers already available to ISMB Researchers.

The results showed that the adoption of this infrastructure has reduced the execution time of 75 percent with respect to the sequential execution on a single machine. Studies are underway to improve the implementation of the scheduler, ensuring that blocks will be distributed more efficiently to the nodes. In particular, a system to take into account the weights of the jobs and the weights of the nodes available will be developed, assigning jobs to specific nodes on the basis of predetermined criteria (size of files, hardware resources, availability nodes, the average time of execution, etc.). The Scheduler will also be able to turn on and off the virtual nodes according to the needs of the system, considering the load level of the grid.

Furthermore the virtual nodes will be sized dynamically as needed to perform specific jobs (requirements for RAM, CPU, storage). This first phase of the activity was aimed at reducing the generation of the entire domain basis functions on each sub block of the original geometry.

A further improvement will be the integration of the system with Public Cloud Platforms (e.g. Amazon EC2) in order to increase the system scalability, asking for virtual nodes usage only when necessary.

Capitolo 4

Cloud Computing for NGS

4.1 Next generation Sequencing Context: NGS

The introduction of next generation sequencing (NGS) technologies to the market has completely revolutionized the functional genomic research leading to an unprecedented availability of biological data. The main novelty consists in the possibility of sequencing an entire genome or transcriptome sample with substantially lower costs respect to the previous Sanger sequencing methodology. The reason behind this biotechnological performance is the main novelty of Next Generation Sequencing (NGS) technologies for capability of NGS machines to chop the DNA/RNA (see Fig. 4.1) molecules into small fragments, namely reads, that are successively sequenced in parallel with considerable saving in terms of time and economic resources.

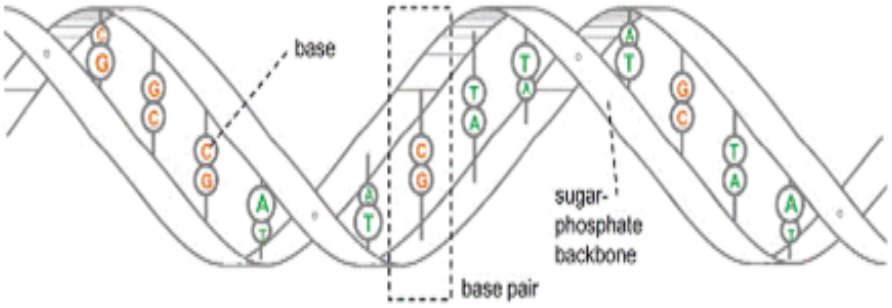

Figura 4.1. The molecular structure of DNA

On figure 4.2 the Illumina sequencing process: (A) Arrays are washed with a solution containing non-specific primers (to start the extension reaction), DNA polymerase enzyme and each of the four deoxyribonucleotides. Temperature cycling

allows just one base extension per cycle. Bases are incorporated into the sequence, complementary to the amplified template. After incorporation, laser excitation causes each base to fluoresce differently and the fluorescence is recorded. (B) The fluorescent tag is cleaved from the last base and a temperature cycle starts. During subsequent cycles, new bases are added and the fluorescence pattern is recorded. (C) The fluorescence pattern is used to construct the sequence at each position on the array. (D) Sequence fragments are aligned to the reference sequence. Where there is discordance to the reference sequence, its frequency is used to determine if the variant is heterozygous (as here) or homozygous. Reference databases are searched to determine whether variants are novel or previously recognized as SNPs.

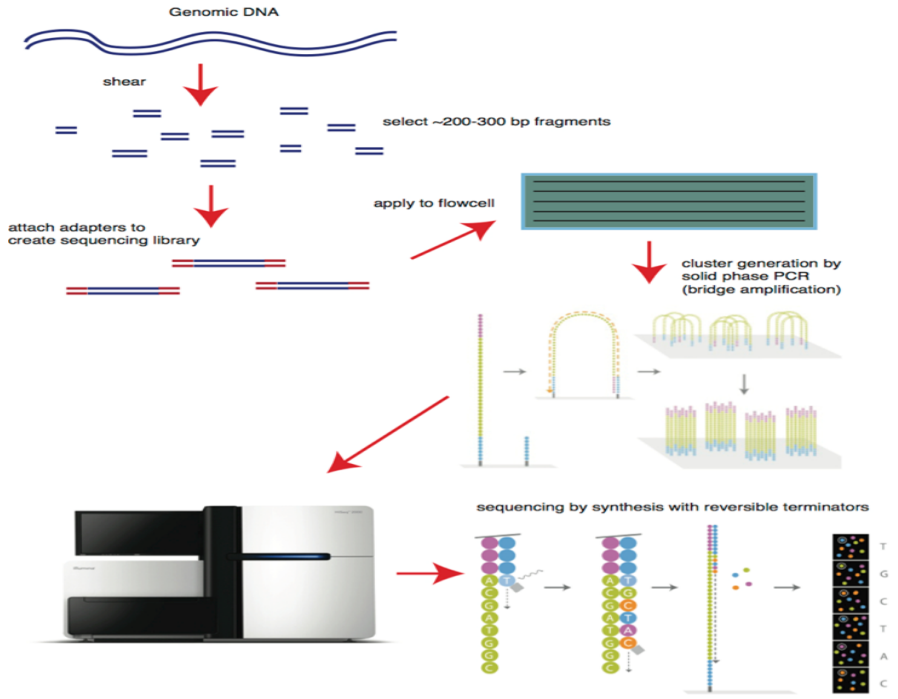

Figura 4.2. Next generation Sequencing

As a consequence, to date more and more biotechnological laboratories are able to produce a huge amount of DNA/RNA sequencing data opening the way to new and more accurate biological analysis such as targeted resequencing, gene expression profiling, small non coding RNA profiling, novel genes discovery, aberrant transcript event detection. These applications heavily impact on the health research field. However, even if from the biological point of view NGS technology leads to new exciting perspectives spreading an incredible amount of data, on the other hand it

raised new challenges in the development of tools and informative infrastructures.

In fact Next Generation Sequencing (NGS) technologies represent a disruptive innovation in the field of functional genomic research leading to an unprecedented availability of biological data. A NGS sequencer can produce millions of reads in a single run that must be then further processed to extract biological knowledge. From a biological and technical point of view NGS technology leads to new challenges in terms of development of tools and computational infrastructures. Computing infrastructures and software tools must be able to handle the huge amount of sequencing data produced by biotechnological laboratories. This capability is key to perform, in a reasonable amount of time, those analysis related to the understanding of biological and pathological processes, such as gene expression profiling, small non coding RNA profiling, novel genes discovery, aberrant transcript event detection.

Mutations in the RNA transcription, as chimeric transcripts, are on the base of various forms of disease and NGS proved to be extremely helpful in making the detection of these events more accurate and reliable. However, even if from the biological point of view NGS technology leads to new exciting perspectives spreading an incredible amount of data, on the other hand it raised new challenges in the development of tools and informative infrastructures. The first step is the reads alignment, i.e. the mapping of the segment on a genome reference in order to reconstruct the original sample sequence and reveal fundamental biological information. Even if the alignment is a very basic operation, due to the great number of data involved in the process, the computational effort in this phase is very high. This scenario recalls for the need of developing computing infrastructures presenting high performances CPU capability and memory availability. For the above reasons grid and cloud based solution might be extremely helpful.

Dealing with a huge amount of data, the more the analysis requires additional and more detailed steps, the more the execution time increases. For instance, the reads alignment is a very basic operation that maps the reads on a genome reference in order to reconstruct the original sample sequence and reveal fundamental biological information. The huge number of reads to be mapped and the possibly large dimension of the genome reference itself make the alignment a not trivial operation.

A more complex analysis flow is required to accomplish splicing detection, which is required to correctly reconstruct transcripts taking into account exon rearrangements (e.g. alternative splicing events), and thus to perform a more accurate alignment and expression profiling. Splicing detection requires to perform run the alignment algorithm in various phases of the analysis. Various algorithms and tools have been proposed to carry out this task.

4.2 Distributed Infrastructure for NGS: Motivations

This huge amount of data produced with NGS technology needs to be analyzed in an acceptable time. One sample consists of more them 80 million of reads (see Fig. 4.3), and it is very hard to process more samples at the same time. In a common scenario composed of a single workstation the waiting time to obtain results from the elaboration of a such large number of reads increase dramatically. Therefore the execution process needs to be optimized by introducing new computational environment. It is worth to be noting that the alignment phase is particularly suitable to be parallelized, since each mapping operation is applied to each read independently on the other read mapping. This feature must be taken in to account when dealing with a cluster of available processors. The first improvement introduced was the concept of grid computing with the goal to parallelize processes and performing more reads analysis at the same time.

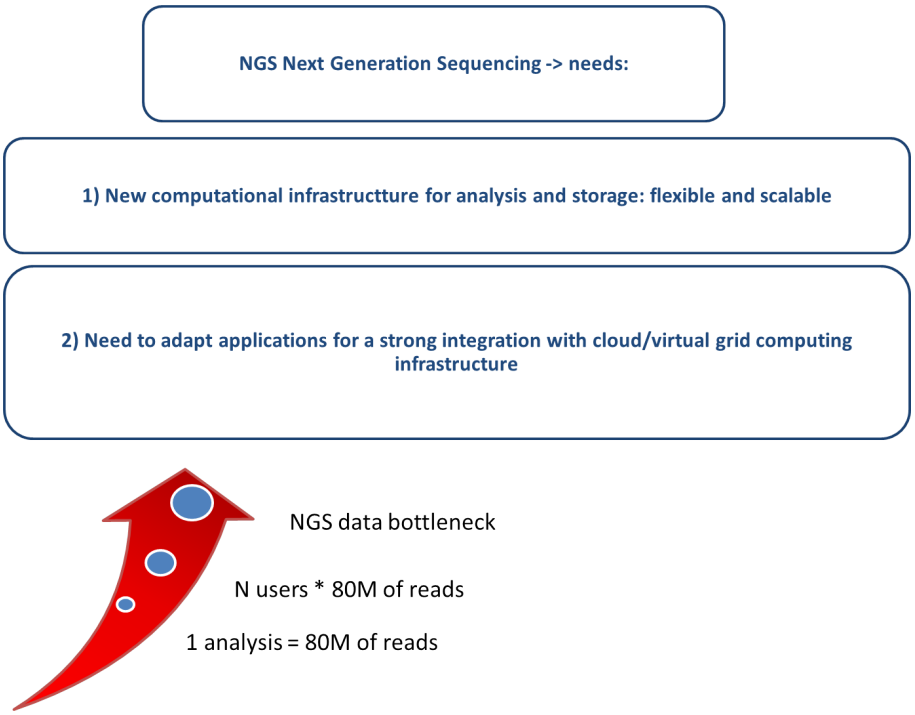

Figura 4.3. NGS needs

Such a complex pipeline imposes tight requirements in terms of processing, memory utilization and storage space not only for input and output data but also for temporary files. Resource utilization is also dependent on the specific execution

phase, thus imposing a dynamic behavior. To address these requirements and promote resource-efficient NGS data processing, parallel software implementations on scalable and flexible computing infrastructures are needed [25]. New implementation of a splicing detection algorithm on a virtualized grid infrastructure which makes efficient use of CPU and memory also enabling a multiuser environment will be propose.

In particular, On this basis an optimization for the structure of the process will be take in consideration, but also the used infrastructure. Therefore for an improvement the concept of Virtualization will be helpfull, that combined with an ad hoc resource scheduler [20], allows to take advantage of a cloudlike environment: for increasing flexibility on the management and introduced the resource optimization.

Thanks to this technology it is possible to host on the same physical machine several virtual machines, that can share the hardware resources and, in case of saturation of the physical resources already allocated, can increase on demand the number of virtual nodes. It is also a multitenancy system with the ability to run in parallel mode, on the same platform, the analysis of different samples from multiple users. More often application are thinking for single user an in monolithic process elaboration.

Main motivations are around two macro domains: a multiuser environment and application parallelization for an improvement of the flexibility and for having good capabilities in term of horizontal and vertical scalability. Moreover with Virtualization it could be created an hardware transparent platform, available on whatever physical machine, speeding up the integration of new infrastructure nodes (using preconfigured templates) and improving the availability of nodes (in case of damage a virtual node will be quickly restored).

4.3 Alignment phase

The short reads alignment is surely the most common operation in RNA-Seq data analysis. The purpose of the alignment is to map each short read fragment onto a genome reference. Each short read consists in a sequence of four possible characters corresponding to the DNA bases and the sequence length depends on the sequencing machine adopted for the biological experiment. The main novelty introduced by NGS technology is the capability of sequencing small DNA/RNA fragments in parallel, increasing the throughput and producing very short reads as output.

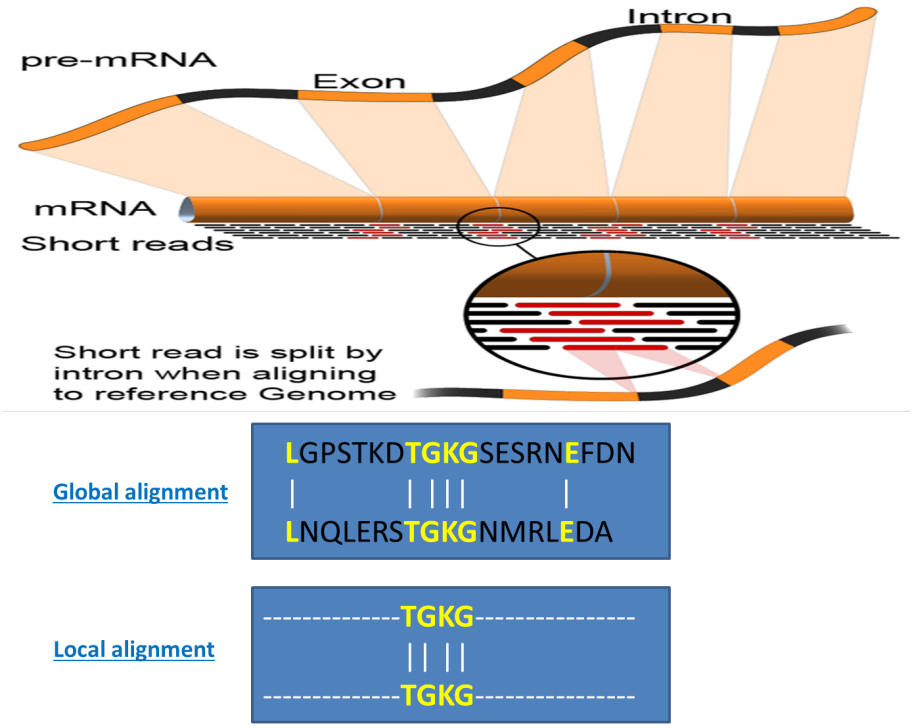

Figura 4.4. Sequence alignement

Inbioinformatics, asequence alignmentis a way of arranging the sequences of DNA,RNA, orproteinto identify regions of similarity that may be a consequence of functional, structural, orevolutionaryrelationships between the sequences.Aligned sequences of nucleotideoramino acidresidues are typically represented as rows within amatrix. Gaps are inserted between theresiduesso that identical or similar characters are aligned in successive columns.

The short reads alignment is very important phase of the entire process since it is the most common operation in RNA-Seq data analysis. Each short read consists

in a sequence of four possible characters corresponding to the DNA bases and the sequence length depends on the sequencing machine adopted for the biological experiment. During the alignment phase each short read fragment is mapped onto a reference genome. It was considered as reference the last human genome produced to now, the HG19 assembly, released in the 2007.

Two steps (see Fig. 4.4):

a Global alignment: Global alignments, which attempt to align every residue in every sequence, are most useful when the sequences in the query set are similar and of roughly equal size

b Local alignment: Local alignments are more useful for dissimilar sequences that are suspected to contain regions of similarity or similar sequence motifs within their larger sequence context.

However, this feature make the computational problem more challenging because of the higher amount of read produced and the accuracy in the mapping (the shorter the sequence length, the higher the probability of having multiple matches). For this reason many alignment tools specifically focused on the alignment of short reads have been recently developed. Bowtie is a wide diffused alignment program particularly aimed at align short reads. In order to detect the actual limitation of the alignment phase.

NGS data coming from the analysis of Chronic Myeloid Leukemia was choose. The samples consist of two file of about 37 million of reads each. The two file are FASTA formatted paired-end reads. Dealing with paired-end reads means that the reads are sequenced by the sequencing machine only on the end of the same DNA/RNA molecule, thus the sequence in the middle part is unknown. Each sequenced end of the same read is also referred as mate. From the computational point of view, it results in two distinct file, the first one consists in the first mate of the same reads and the second one consists in the opposite mate. As previously aforementioned, an alignment program maps each read on a reference genome.

The HG19 assembly produced in the 2007 is considered as reference genome the last human genome assembly produced to now. The output of Bowtie consists of three data files:

1 the SAM file format containing the set of aligned read,

2 FASTA file containing the set of reads that Bowtie failed to align,

3 FASTA file containing the list of read that maps more than a specified number of time on the genome reference. Specifically, the SAM format specifies for each read all the information needed to completely

describe the way the read maps on the reference genome (see [15] for further specifications).

Each short read consists in a sequence of four possible characters corresponding to the DNA bases and the sequence length depends on the sequencing machine adopted for the biological experiment. The main novelty introduced by NGS technology is the capability of sequencing small DNA/RNA fragments in parallel, increasing the throughput and producing very short reads as output.

The short reads alignment is surely the most common operation in RNA-Seq data analysis. The purpose of the alignment is to map each short read fragment onto a genome reference.

4.4 Tophat algorithm and Bowtie alignment tool

4.4.1 Bowtie

For instance, reads alignment is a basic operation that maps the reads on a genome reference in order to reconstruct the original sample sequence. This step is used as building block of more complex analysis pipelines, such as for alternative splicing or gene fusion detection.

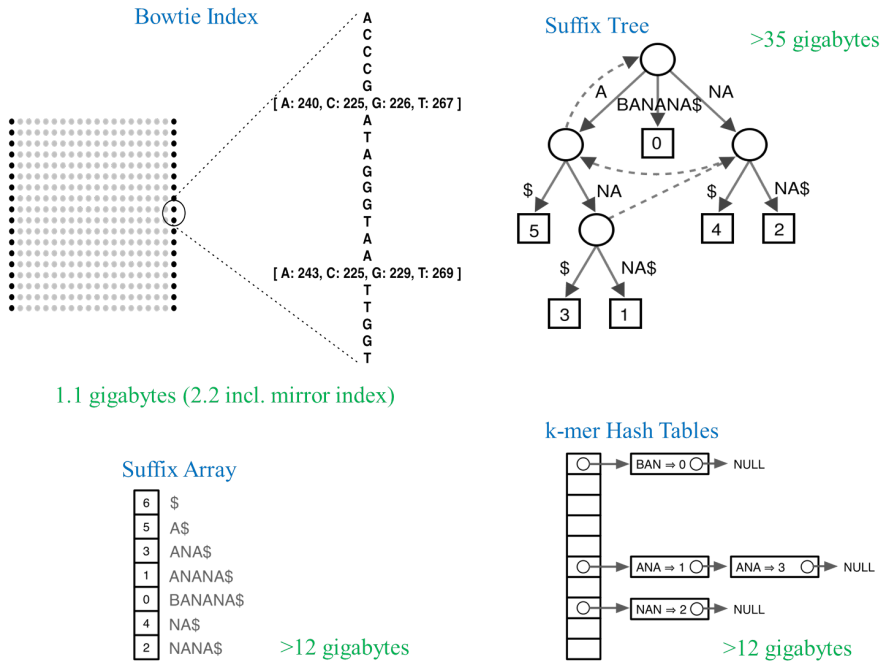

Figura 4.5. Bowtie indexes

However, due to the large number of reads, this basic block becomes critical. Several algorithms and the associated tools have been recently developed to optimize the alignment phase, in particular Bowtie is widespread because of its effectiveness when dealing with short reads (50-100bp) with respect to previous solutions. Moreover, it supports multi-threaded processing and it makes an efficient usage of system memory.

Bowtie is an ultrafast, memory-efficient short read aligner. It aligns short DNA sequences (reads) to the human genome at a rate of over 25 million 35-bp reads per hour. Bowtie indexes the genome with a Burrows-Wheeler index to keep its

memory footprint small: typically about 2.2 GB for the human genome (2.9 GB for paired-end). Bowtie extends previous Burrows-Wheeler techniques with a quality-aware search algorithm that permits mismatches. Multiple processor cores can be used simultaneously to achieve greater alignment speed. Bowtie is free, open source software available for download from http://bowtie.cbcb.umd.edu.

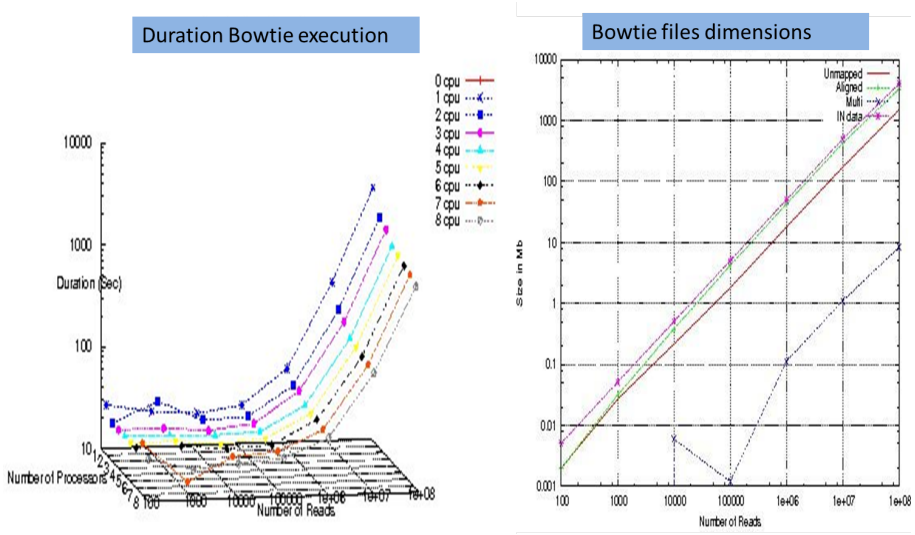

Figura 4.6. Bowtie execution

Bowtie builds a genome index (see Fig. 4.5) based in the Burrows-Wheeler Transformatopn (BWT)2 and FM Index3. The Burrows-Wheeler Transformation of a text T, BWT(T), is constructed as shown to the right. The Burrows-Wheeler Matrix of T is the matrix whose rows are all distinct cyclic rotations of T$sortedlexicographically$(is less than all other characters). BWT(T) is the sequence of characters in the last column of this matrix.The Burrows-Wheeler Matrix has a property called the LF mapping: the ith occurrence of character X in the last column corresponds to the same text character as the ith occurrence of X in the first column. This property underlies algorithms that use the BWT to navigate or search the text.

An index for T consists of BWT(T) and some auxiliary data structures. For efficiency reasons, Bowtie indexes both the genome (the forward index) and its reverse (the mirror index). For the human genome, the total size of the Bowtie is smaller than other popular indexing schemes:

In a learning phase, some tests were performed in order to make the best architectural choice. We introduce two case studies from two different point of view. The

first is the fragment type, while the other one is the CPUs number on the worker node. On figure 4.6 a summary of the calculations obtained changing CPUs number are presented. As we can notice for reads between 100 and 1000000 no gain of time has occurred, so for our studies only reads from 1000000 to 85000000 are considered.

In order to increase the computational performances during the read mapping, Bowtie program creates an index of the provided human genome reference. This operation is particularly straightforward from the computational point of view, but it must be performed only one time for the human genome reference and it is independent on the mapping samples. The alignment phase itself is particularly suitable to be parallelized. In fact, each mapping operation is applied to each read independently on the other read mapping.

Some tests were conduct for better understanding the CPU context switching base on the number of CPU available on the machine and the thread number indication during tests. on figure a test conduct by using 5 thread for Bowtie execution on a computational resources with 8 core.

During the execution there is a frequent switching between all core available on the hardware, on each instant 5 core are used but not all the time the same core or group of core was used (see fig. 4.7).

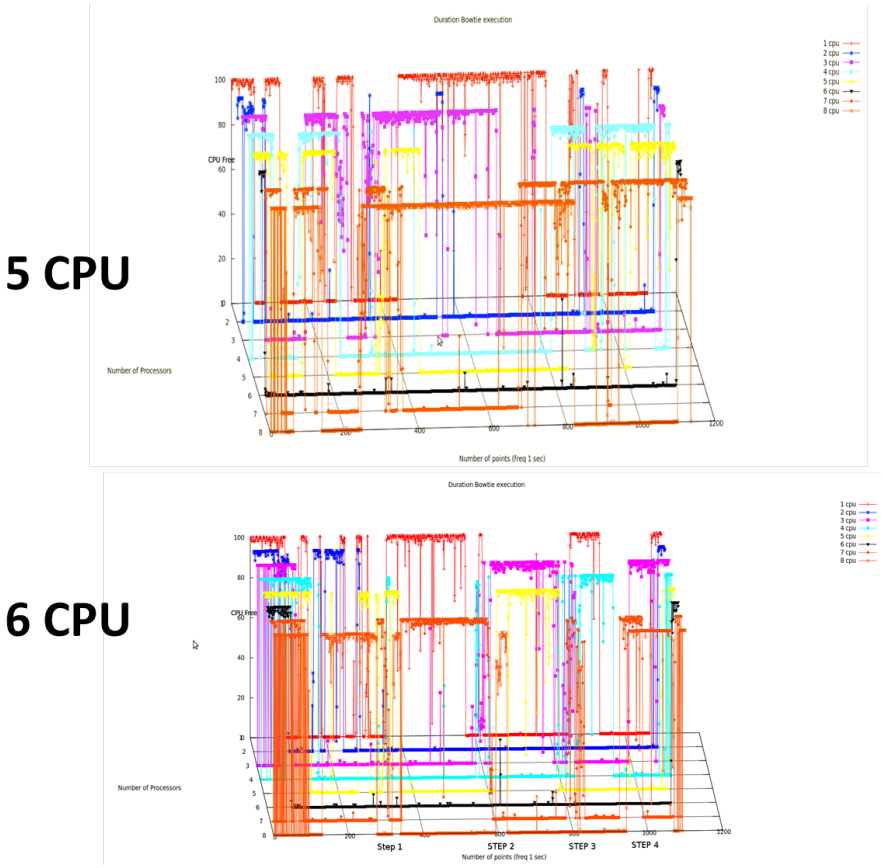

Figura 4.7. CPU switching

4.4.2 Tophat algorithm

TopHat is a fast splice junction mapper for RNA-Seq reads (see fig. 4.8). It aligns RNA-Seq reads to mammalian-sized genomes using the ultra high-throughput short read aligner Bowtie, and then analyzes the mapping results to identify splice junctions between exons. TopHat is a collaborative effort between the University of Maryland Center for Bioinformatics and Computational Biology and the University of California, Berkeley Departments of Mathematics and Molecular and Cell Biology.

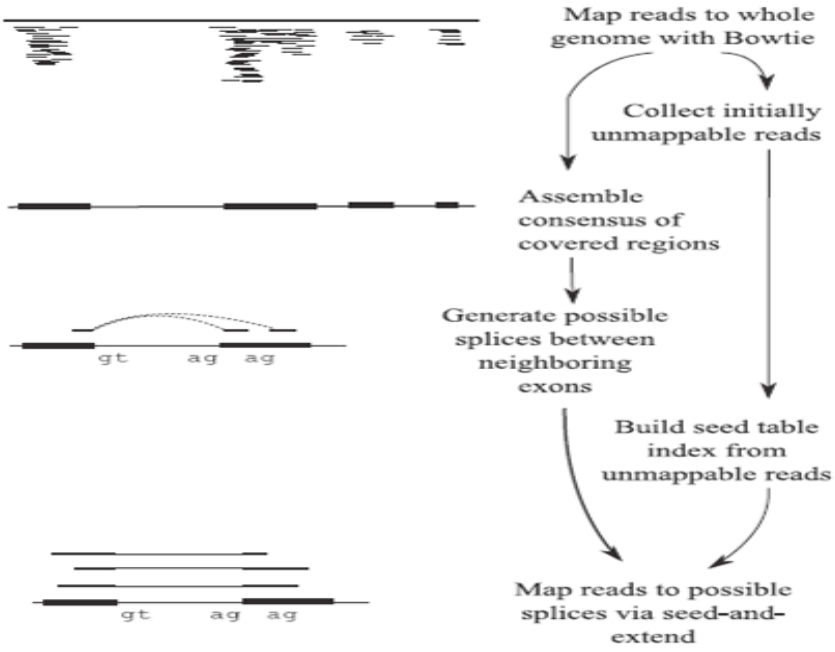

Figura 4.8. TopHat: RNA-Seq reads

TopHat receives as input reads produced by the Illumina Genome Analyzer, although users have been successful in using TopHat with reads from other technologies. The input samples consist of two file of about 37 million of reads each. The two file are FASTA formatted paired-end reads. Dealing with paired-end reads means that the reads are sequenced by the sequencing machine only on the end of the same DNA/RNA molecule, thus the sequence in the middle part is unknown. Each sequenced end of the same read is also referred as mate. It results in two distinct files, the first one consists in the first mate of the same reads and the second one consists in the opposite mate.

TopHat finds junctions by mapping reads to the reference in two phases. In the first phase, the pipeline maps all reads to the reference genome using Bowtie. All reads that do not map to the genome are set aside as initially unmapped reads. Bowtie reports, for each read, one or more alignment containing no more than a few mismatches in the 5-most s bases of the read. The remaining portion of the read on the 3 end may have additional mismatches, provided that the Phred-quality-weighted Hamming distance is less than a specified threshold.

TopHat is build on top of Bowtie and it is specifically aimed at finding new splice junction and mapping reads on top of them. The importance of finding new junctions are: to improve knowledge of trascriptional biological process and to improve reads mapping probability. Consequently, it performs a more accurate alignment. TopHat receives as input reads.

This policy allows so called multireads from genes with multiple copies to be reported, but excludes alignments to low-complexity sequence, to which failed reads often align and then assembles the mapped reads. TopHat extracts the sequences for the resulting islands of contiguous sequence from the sparse consensus, inferring them to be putative exons. TopHat produces a compact consensus file containing called bases and the corresponding reference bases in order to generate the island sequences.

TopHat uses the reference genome to call the base. Because most reads covering the Alignment Phase. ends of exons will also span splice junctions, the ends of exons in the pseudoconsensus will initially be covered by few reads, and as a result, an exons pseudoconsensus will likely be missing a small amount of sequence on each end. In order to capture this sequence along with donor and acceptor sites from flanking introns, TopHat includes a small amount of flanking sequence from the reference on both sides of each island. TopHat has a number of parameters and options, and their default values are tuned for processing mammalian RNA-Seq reads also it can be used for another class of organism.

Study for parallelization of some TopHat workflow analysis in order to find an optimal process scheduling in a flexible computational infrastructure for the main execution steps of TopHat have been explored.

4.5 Tophat reverse engineering: Motivations

In a preliminary phase of reverse engineering of TopHat [24, 23], blocks of transactions have been identified that were executed sequentially. We have divided TopHat flow into three steps, that are executed by Bowtie. We have identified three main blocks(see Fig. 4.9), that can be executed independently:

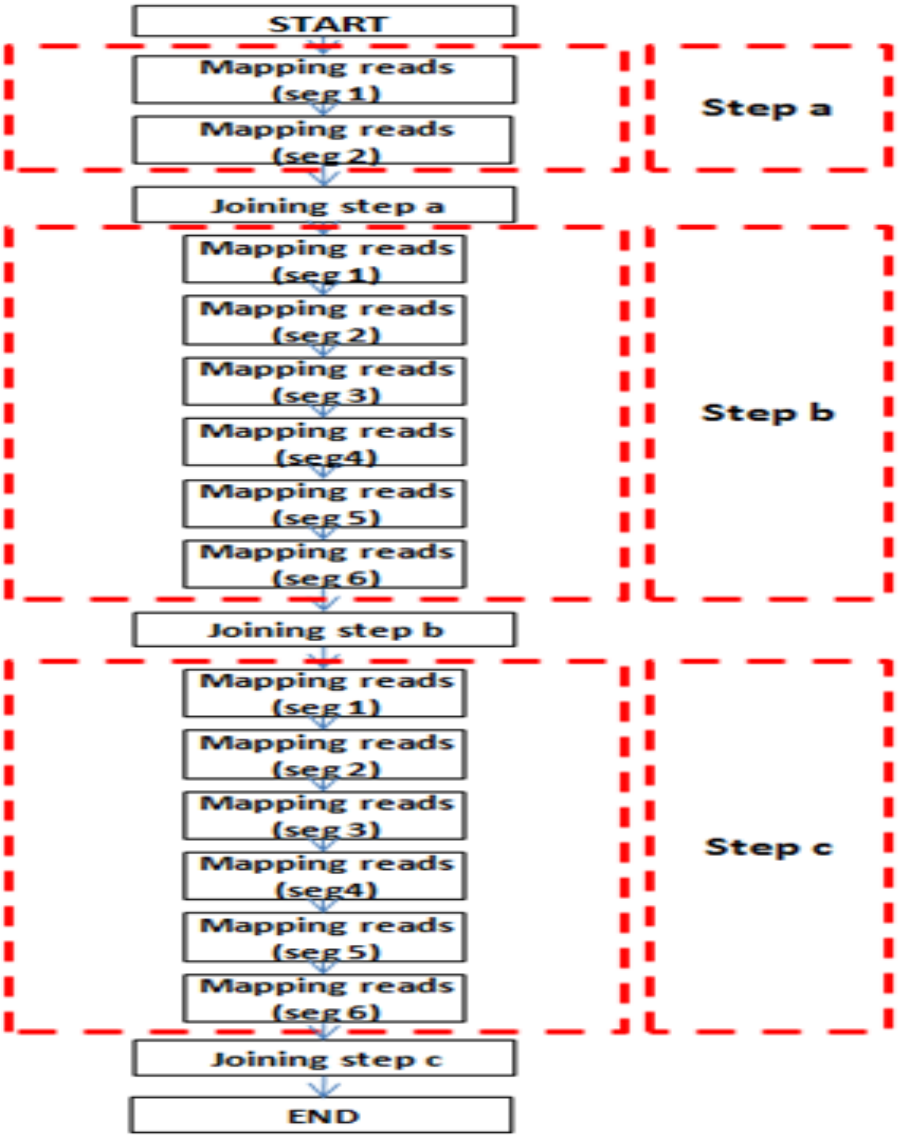

Figura 4.9. TopHat Native Version

a Left and right check reads segments;

b Left and right mapping segments with HG19;

c Left and right mapping segments with segment juncs.

At the end of each step, there is a join phase of the results previously obtained. For Steps (a) and (c), since files involved in the elaboration are significant, a common repository has been created that contains the temporary folder used by TopHat. On Figure 4.9 Simple Sequential TopHat Flow. The input sample consists of two files of about 37 million of reads each. The two files are FASTA formatted paired-end reads. Dealing with paired-end reads means that the reads are sequenced by the sequencing machine only on the end of the same DNA/RNA molecule, thus the sequence in the middle part is unknown. Each sequenced end of the same read is also referred as mate.

NGS data sample consists of millions of reads, and in a classic situation, with only one workstation available, time needed to obtain the output increases significantly. Alignment is a process in which each mapping reference is made to read independently from the other reads, and this means that can perform a parallel analysis of the data. Although alignment is a very basic operation, computational effort is very high that is due to the great number of data involved in the process.

In NGS technology context the amount of data and the number of samples to be analyzed is growing constantly.

It is a positive factor for increasing more accurate studies and results in term of reliable identifications of mutations in aberrant splicing events, fused genes and open new perspective and challenges for adapting tools that are in condition to make pre and post processing in modern infrastructure like virtualized machine, Grid and Cloud computing.

A NGS data sample consists in a millions of data and the time needed for the process execution increase dramatically with only one workstation for processing NGS data. The alignment phase is a process which each mapping reference is made in independent way and can be execute in a parallel way on a distributed computing context. The alignment is a very basic operation but actually the alignment tool that we use, TopHat, provides a sequential analyze of each block of reads and in a single user way and consequently in a typical scenarios were more samples need to be analyze the total time for processing data is not acceptable. In consideration of this a first challenge is to adapt the tool by making a reverse engineering of the code for a transformation of all possible main and heavy sequential alignment process and to a parallelization way in order to obtain an optimization of process time.

After a reverse engineering for a better understanding of the tophat native version, a second challenge was to adapt and change the native version by a parallelization of the all sequential analysis. The main idea was to eliminate all loop and for piece of code that involve a sequential analysis. Tophat is write in python language so some change was implemented by inserting command for a distribution of all segments for a bowtie alignment on order to begin on a parallel way the bowtie process, on figure 4.10 an example of change inside the tophat source file.

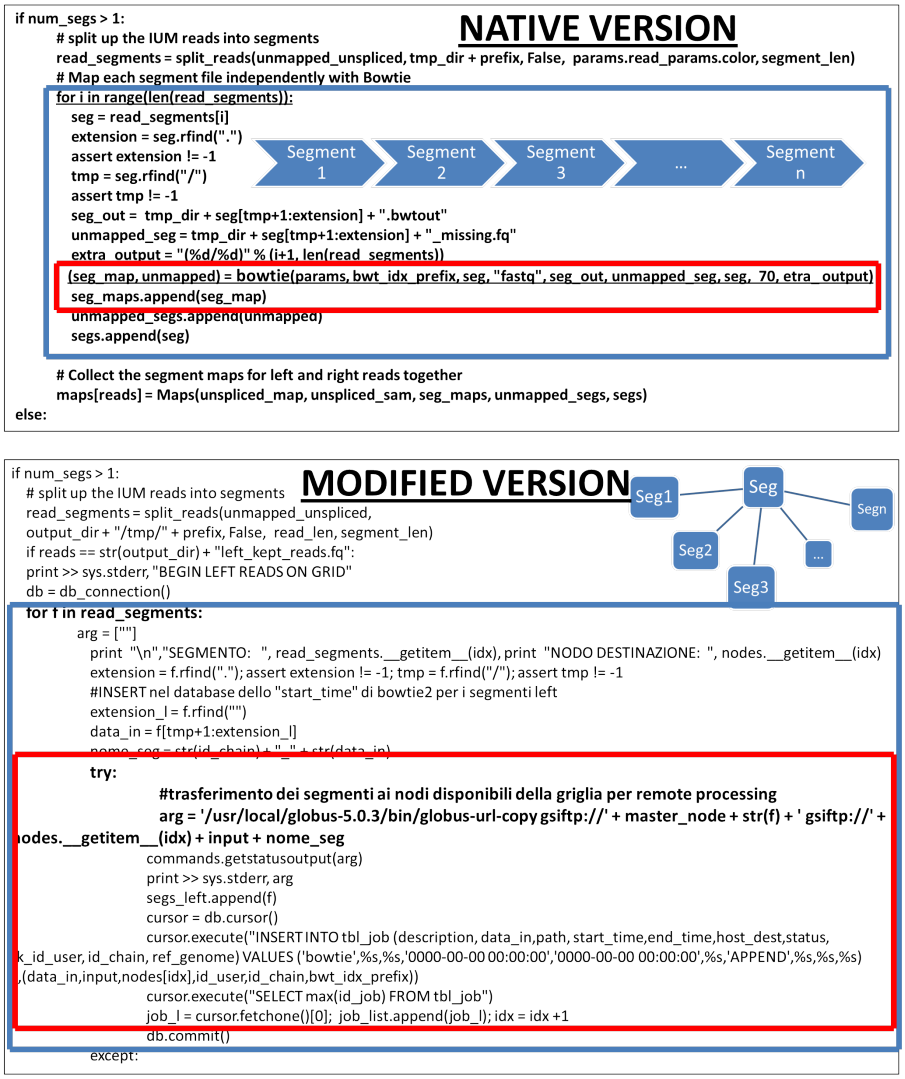

Figura 4.10. Source file modification

The alignment tool for an multi sample environment. This new scenario characterize the capabilities to make splice junction mapping in an flexible and distributed computing infrastructure.

On figure 4.11 the new flow of a parallelized version of tophat.

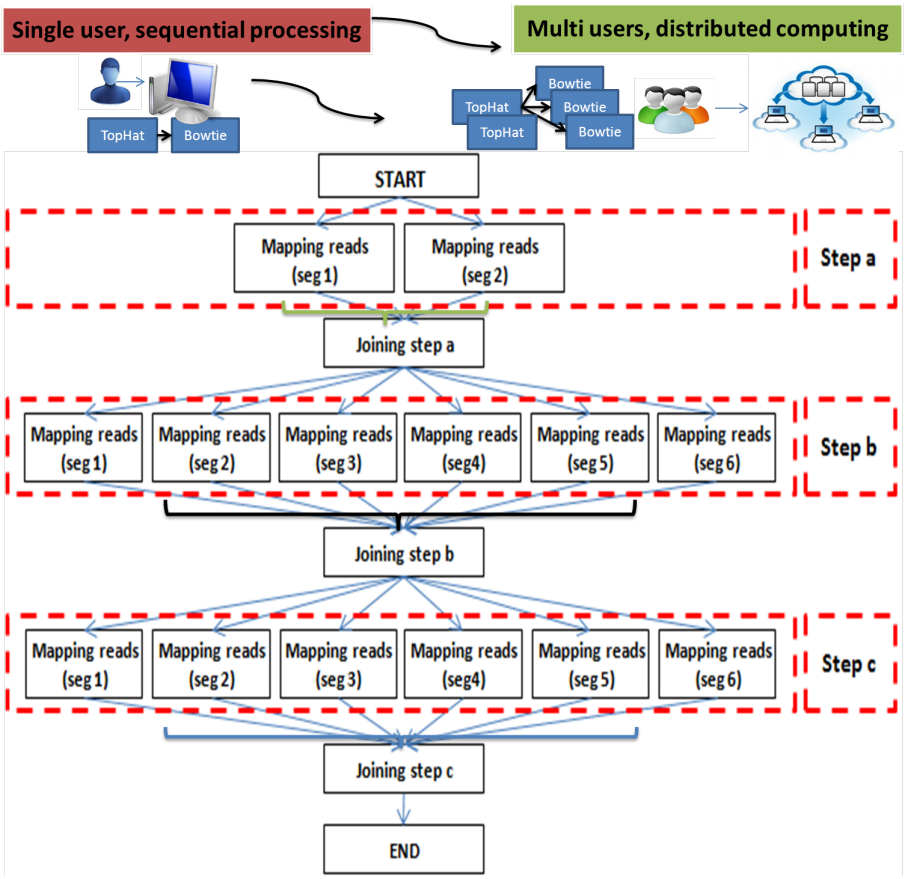

Figura 4.11. Tophat parallelized version

4.6 NGS Virtual BIO Distributed infrastructure

Therefore, the main challenges in the elaboration of NGS data can be reconducted to the elaboration of million of reads for each single sample. Due to this consideration some specific challenges for news scenarios are focused on developing new computing infrastructures by adapting biological tools for a integration in virtualized grid infrastructure (see Fig. 4.12) with high performances CPU capability and memory availability in a multi user context.

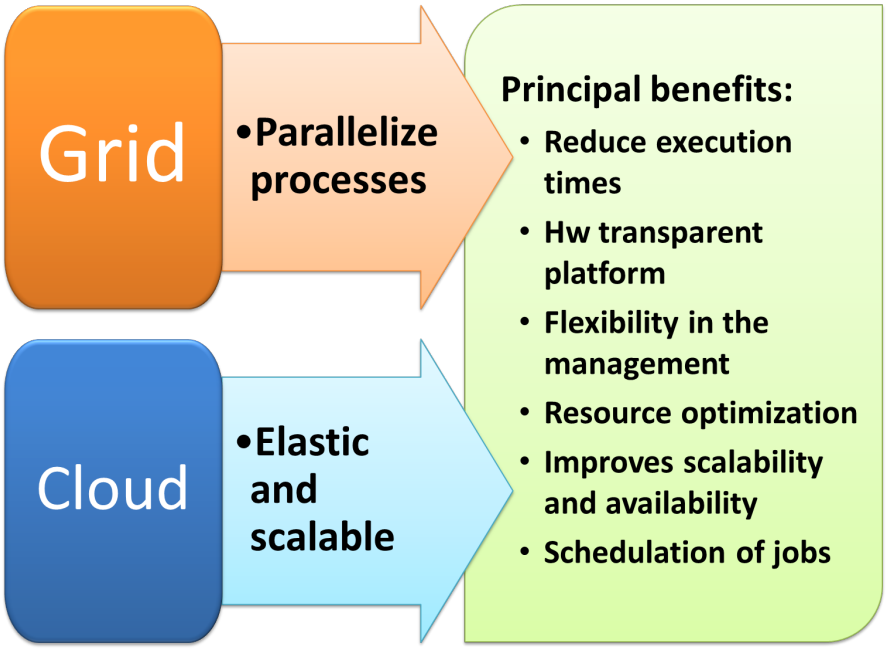

Figura 4.12. Grid and Cloud integration

However, an accurate analysis on how optimally integrate NGS data on a grid based system is performed [26]. The nature of NGS data makes this aspect not trivial because of the tradeoff between the data transferred on the network and the improvement of flexibility for a multi user context. This scenario recalls for the need of developing computing infrastructures presenting high performances CPU capability and memory availability. The grid environment consists of machines with high computing power. Grid environments are scalable, making them effective for uses where storing large amounts of data.

The only requirement is to have the necessary software installed for the processing (Bowtie and Globus). On each worker node of the grid environment is installed

the Grid Local Scheduler, an essential component for performing biological tests.

An hybrid infrastructure, called NGS Hybrid [25], was created composed of two different parts. The first one is a distributed architecture composed of a set of physical nodes called NGS Grid and the second one is a set of virtual nodes in a context of Private Cloud Computing called NGS Cloud that provides a dynamic scalability of the infrastructure, Grid and Cloud technology for an improvement of the computational infrastructure. A fundamental component of the system is the Master Node, which provides the management of resources and the scheduling processes.

Workers Nodes are in charge of the execution of a part of the whole process. The system relies on the middleware Globus Toolkit, that allows to share computing power and other resources securely online. To setup the grid computing environment, the system is based on the middleware Globus Toolkit, for a variety of reasons. The open source Globus Toolkit is a fundamental enabling technology for the grid, letting people share computing power, databases, and other tools securely online.It provides all the functionalities needed to implement a seamless grid layer that allows faster analysis of RNA sequences by harnessing workstations and clusters.

Globus toolkit includes software for security, information infrastructure, resource management, data management , communication, fault detection, and portability. It is packaged as a set of components that can be used either independently or together to develop applications.

Globus middleware is a common layer for both grid and cloud infrastructure, and the nodes of both belong to the same Virtual Organization (VO). The pool of resources on which the infrastructure is relying consists two hardware platforms: three 8 cores workstation with 1.60 GHz Intel Core i7 processor and 8GB of RAM, for the grid architecture, and a large-memory server with 12 cores 2.80 GHz Intel Xeon processor and 32 GB of RAM devoted to the cloud platform.

As hypervisor the open source KVM has been used, which allows creating fully-virtualized virtual machines (VM), running an unmodified kernel. The basic requirement for the installation of this hypervisor is that the processor of the machine has virtualization support (Intel VT or AMD-V). Virtualized environment [22] also helps to improve infrastructure management, allowing the use of virtual node template to create virtual nodes in a short time, speeding up the integration of new nodes on the grid and, therefore, improving the reactivity and the scalability of the infrastructure.

The kernel component of KVM is included in mainline Linux. KVM allows a

Full Virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). KVM is implemented as a module within the Linux kernel. A hypervisor hosts the virtual machine images as regular Linux processes, so that each virtual machine image can use all of the features of the Linux kernel, including hardware, security, storage and applications. Full Virtualization provides emulation of the underlying platform on which a guest operating system and application set run without modifications and unaware that the platform is virtualized.

It implies that every platform device is emulated with enough details to permit the guest OS to manipulate them at their native level. Moreover, it allows administrators to create guests that use different operating systems. These guests have no knowledge about the host OS since they are not aware that the hardware they see is not real but emulated. The guests, however, require real computing resources from the host, so they use a hypervisor to coordinate instructions to the CPU.

The main advantage of this paradigm concerns the ability to run virtual machines on all popular operating systems without requiring them to be modified since the emulated hardware is completely transparent. The virtualized environment has pre-installed images, which contain all software and libraries needed for running Bowtie and TopHat. The pre-configured images allow an ease instantiation of the machines when needed, and can be easily shutdown after the use.

The operating system running onto each node, both Master and Worker (physical or virtual), is a Linux Ubuntu 10.04 x86 64. Moreover each worker node is equipped with the grid services provided by Globus Toolkit, a local Scheduler for the job execution, the Bowtie tool for alignment and the indexed files of the HG19 human genome.

NGS Hybrid Workflow [19] virtual node was created including all these software and integrated within the same VO of NGS Grid. It can be reused also to instantly start up virtual nodes. The Master Node hosts the Global Scheduler and the central database since it is in charge to monitor and assign job to the most suitable worker node belonging to cloud or grid infrastructure.

A MySQL database was developed to contain information about jobs (identifiers and logs of each process), nodes (status and availability) and the requests of resources to NGS Cloud infrastructure. Globus services are used to submit the jobs to worker nodes, to send executable and data files to computing resources and to retrieve the computation results. The monitoring is ensured by a mechanism of running agents, a solution that provides agility to the system and improve the scalability. Each node owns its agent, that periodically communicates with the Master

Node. The availability of each service on the node means that the node is able to execute jobs. These information are updated into the central database.

The NGS-Hybrid [20, 21]is an infrastructure (see figure 4.13) devoted to NGS data analysis, and it is mainly focussed on the alignment of short sequence reads on the reference genomes. Each worker node has installed the same software, namely Bowtie, in order to be able to execute the alignment job. Moreover, each node is equipped with the indexed files of the HG19 human genome reference. As aforementioned, the operation of creating an index of the reference is performed only one time for genome reference, therefore this fixed time can be neglected in the performances of the alignment phase.

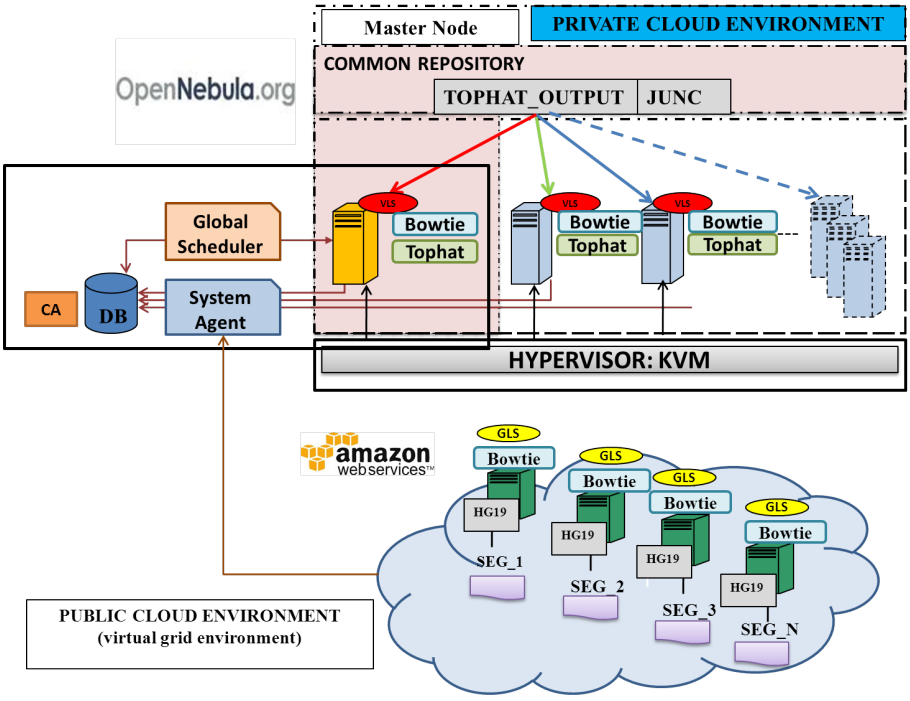

Figura 4.13. NGS Hybrid Cloud Infrastructure

Globus handles all the details of sending executable and data files to computing resources and retrieving the computation results. Each job is identified by a unique identifier, it runs as command-line bash script and Globus is used to submit the jobs to worker nodes. This job runs the Bowtie execution. Globus also provides the functionality needed to check whether a submitted job has completed execution: this functionality is blocked while the job is running and only returns when the job

is completed.

Virtualized environment also helps to improve infrastructure management, allowing the use of virtual node template to create virtual nodes in a short time, speeding up the integration of new nodes on the grid and, therefore, improving the reactivity and the scalability of the infrastructure. The virtualized environment has pre-installed images, which contain all software (Bowtie and TopHat), local schedulers (VLS, GLS) and support data (HG19). To have images already configured allows to set up easily machines when you need them, and once used the close the instance.

An implementation of a parallelized and modular version of TopHat was integrate on the distributed infrastructure, going beyond the inherent multithreading support of Bowtie, supporting multiple parallel sample processing and performing a careful evaluation of computation versus communication trade-off, which is critical when large data masses have to be transferred. Network File System (NFS) has been used to configure the common repository (CR), that allows computers to share files and folders over a network. Step (b) instead involves small files, these can be performed on a Grid, both physical and virtual, because transfer time is neglected.

The samples used during the tests are NGS data coming from the analysis of Chronic Myeloid Leukemia, collected in two files FASTA formatted pairedend reads. This means that each read is sequenced by the sequencing machine only on the end of the same DNA/RNA molecule and the sequence in the middle part is unknown. The sequenced end of the same read is also referred as mate.

Bowtie is a widely diffused alignment tool, it is more efficient with short reads respect to previous solutions and supports multi-threaded processing with an efficient memory usage. It can parallelize the alignment process using the pthreads library, allowing the user to specify a desired number of threads and running the search functions along concurrent threads. Thanks to this technique, it is possible to speed up alignment on computers with multiple processor cores.

The memory image of the index is shared by all threads, and so the footprint does not increase substantially when multiple threads are used. In the infrastructure, each parallelized TopHat instance runs to process a single sample. Many samples processing can be interleaved and the scheduler allocates the various execution steps on the physical resources, namely the worker nodes, to minimize execution time. The infrastructure leverages upon the integration of a hybrid and flexible computing infrastructure composed by physical workers nodes in a common standard grid

computing architecture with virtual grid nodes in a cloud computing infrastructure.

Two schedulers have been developed: Global and Local scheduler. The Global Scheduler, installed on grid master, allows to collect all jobs to elaborate and to distributing them among the worker nodes. Instead, the Local Scheduler is installed on each worker nodes and its aim is to process data in accordance with the grid master request. Experiments carried out to characterize the performance of the proposed solution show the effectiveness in having a multiuser environment and reducing the execution time with respect to a standard version of TopHat and highlight the improved resource utilization allowed by the virtual infrastructure.

4.7 Scheduling

Three job schedulers have been developed: Grid Local Scheduler (GLS), Virtual Local Scheduler (VLS) and Global Scheduler (GL). GLS has been developed for Step (b), is active on physical machines, it aligns segments with respect to the HG19 through Bowtie. Since transfer of input file can be neglected, Worker Nodes do not need to be in the same subnet, but may also belong to different virtual organization, so system can have greater scalability and can use machines powerful performance. VLS is a scheduler active on virtual machines.

Its purpose is to draw up Steps (a) and (c) through Bowtie but Step (a) allows alignment with respect to the HG19 and Step (c) allows alignment with respect to the segment juncs previously constituted by TopHat. Since the considerable size of the files involved in these 2 steps, VLS works directly in the temporary folder that is located in the RC, allowing to avoid wasting time due to the data transfer. GL is located on MN, it checks nodes available in the Grid infrastructure, when not enough nodes are available, it provides to set up virtual machine and distributes input file to the Worker Nodes, then it waits to receive all output files to proceed with the next step of the flow.

It was implemented a dynamic infrastructure that can automatically provide the upscaling and the downscaling of system resources. This capability is provided through the Global Scheduler that knows different information about each node of the grid: the grid services status, if they are in condition to receive and execute jobs and if nodes are in running or in waiting mode. The Global Scheduler receives job directly from users and sends executions with priority to NGS Grid nodes until all these resources are busy. When all the physical nodes are executing jobs, the Global Scheduler sends a request for new nodes to the NGS Cloud infrastructure by

updating the central database.

The hypervisor, that manages the virtual resources, has been provided with a daemon responsible to periodically query the database: if there is a request for VMs, the number of worker nodes needed for the elaboration are created. At the same way when the system load is decreased, the Global Scheduler informs the hypervisor that a specific VM could be shut down. Also this time information is receipt from the NGS Cloud daemon through the database. Obviously, the hypervisor before turning off the machines ensures that all processes have been completed.

The analysis workflow consists of three steps: (1) the splitting of the reads, (2) the running of Bowtie, (3) the reassembly of data. The first step is accomplished by the Global Scheduler: it checks if the files to be analyzed are available and splits these in smaller files. It is worth noting, that the operation of dividing the original file of reads into smaller file is possible and it does not affect the results, because each single read is elaborated independently. For the second step, has been developed a Local scheduler, located on every worker nodes which principal tasks are: to wait a new input file, validate it, run Bowtie, return the outputs and clean the workspace. Finally, in the last step the Master Node reassembles the output files containing the aligned reads coming from each worker node in order to complete the analysis.

The Grid environment consists of machines with high computing power, this allows to use own machines and also machines belonging to different virtual organization. The only requirement is to have the necessary software installed for the processing (Bowtie, TopHat and Globus Toolkit). On each worker node of the grid environment is installed the Grid Local Scheduler, an essential component for performing biological tests. 1) Grid Local Scheduler: The Local Grid Scheduler (GLS) is a scheduler active physical machines, has been developed for the design phase (b), it aligns the segment with respect to the human genome (HG19) through Bowtie. Since the transfer of the input file is not influential, the worker nodes do not need to be in the same subnet as the Master Node, but may also belong to different virtual organization, so system can have greater scalability and can use machines powerful performance.

4.8 Experimental results

The purpose of the tests is to identify the best tradeoff between allocated VMs and concurrent threads in order to achieve the greatest alignment speed with this kind of infrastructure. A huge amount of measurements of the alignment time using different number of virtual machines hosted on the same physical machine were made. In order to speed up the execution, the test were performed with an optimized file derived from the real sample, suitable for following biological analysis. This file contains data only from one mate of the sample for an overall number of reads of about 20 millions. The various configurations adopted on the server used to host the cloud infrastructure are: one virtual machine with 12 cores; two virtual machines with 6 cores; three virtual machines with 4 core.

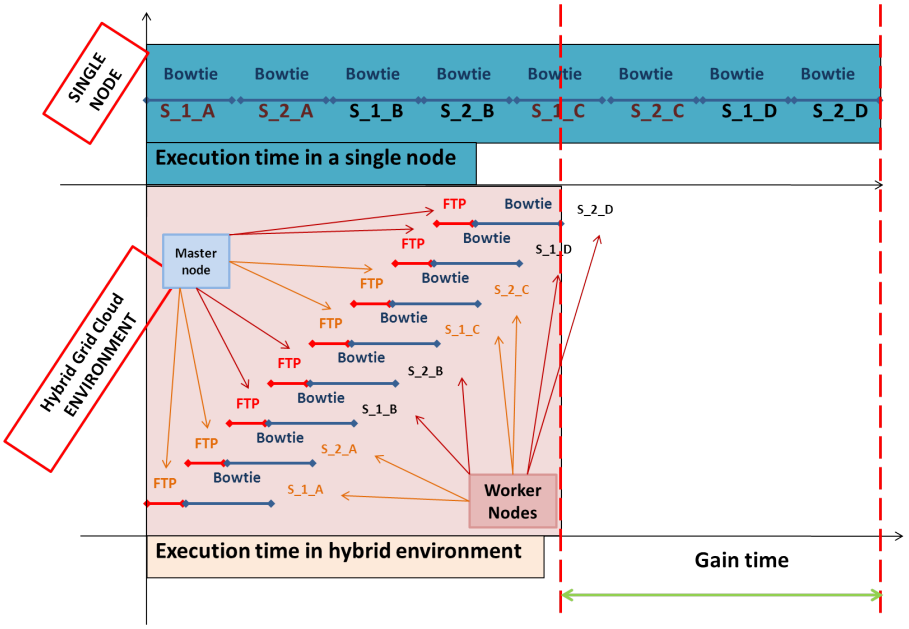

Figura 4.14. Single, NON single node: Execution Time Gain

In the case of one VM the best performance is obtained when Bowtie runs executions in parallel on all available cores (12 cores in this case), if two VMs are allocated the optimal number of threads running on each machine is 6, in the case of three virtual machines is 4. If this bound is exceed the performance decline. the proposed infrastructure introduces an overhead due to the virtualization layer. This overhead is appreciable from the comparison between the physical machine times and the configuration with one virtual Alignment.

It is important to note that the gap between physical and virtual machine is almost constant with respect to the configuration. If it is considered the times corresponding to the optimal number of threads in executions the gap is also constant for all the configurations.

The sequential execution was performed on the physical machine with a number of threads equal to 12. To make the times comparable it was measured the execution time of two and three sequential running of Botwie on the physical machine. Parallel execution was performed on the three configurations with a number of threads equal to the maximum number of physical cores available on each virtual machine.

Input in Step (a), consists of two files and have size about 3 GB each, in Steps (b) and (c), consists of twelve files about 500 MB each instead support files (HG19 and segment juncs) have a size of about 4 GB each. In the original version of TopHat, for elaboration of a single sample, RAM required is about 8 GB and at least 60 GB of free space hard disk.

The aim of tests performed is to compare execution time for multi samples using the two version of TopHat: the original version that sequential approach and version modified exploiting the distributed environment. For this test phase, we wanted to use an architecture which consists of six machines with four CPUs. Original Tophat vs Tophat Grid. savings of 40 percent is obtained instead increasing the number of samples to be processed, it is worth noting that the percentage of earned time is about 30 percent, this is due to the jobs queues that are created on the nodes.

In the analysis of four samples we get 2 packets (S1,S2) of 4 GB for each and the total execution time using only one node is 8320 seconds , about 139 minutes. Instead, in Hybrid environment the total execution time is 4000 seconds, about 67 minutes. We want to emphasize that time gain is already verified by assaying only four samples (see Fig. 4.14). Even if in grid environment, transfer time must be considered, this approach allows to save time of 4320 seconds, about 72 minutes.

Capitolo 5

Conclusion

The increasing number of data and applications in eScience produced needs to be process in a high performance computing infrastructure context in order to make analysis and results in a reasonable time. Hybrid infrastructure proposed gives the capability to auto scale the available resource depending on the number of processes to be analyzed. The novelty of the proposed approach is that the system is based on an hybrid infrastructure that exploits the benefits of grid and cloud computing technology: the ability to parallelize the process of grid platforms and the flexibility provided by the cloud.

The infrastructural solutions proposed for the three use cases cover the field of the optimization of softwares for an integration in Hybrid distributed infrastructure like virtual grid and Cloud Computing environment and the optimization of the use of resources thanks to scheduling models.

Cloud infrastructure drastically improves overall system performances by providing the scalability of the system with the inclusion of virtual nodes when necessary. In this way, the user not need to possess the entire infrastructure but only a few machines, and if necessary it can contact a third-party suppliers to meet the need of temporary computing power. The hybrid model for users will also be a way to save money, pay the service in accordance with the pay-per-use model.

In some cases like NGS and CEM, algorithms has been optimized making parallel independent sections that were sequential and has been modified for giving a multi user environment were before was for a single user instance. In the other case radio occultation the hybrid infrastructure Grid, private and public cloud was suitable for the capabilities to give a scalable solutions.

Bibliografia

[1] O. Terzo, L. Mossucca, P. Ruiu, G. Caragnano, K. Goga, R. Notarpietro and M. Cucca, Improving Scalability of an Hybrid Infrastructure for E-Science Applications, LNCS Transactions on Computational Collective Intelligence, Springer 2012.

[2] Terzo O., Mossucca L., Ruiu P., Caragnano G., Hybrid architecture for intensive data analysis, Sixth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing(3PGCIC2011) Barcellona, Spain, October 26-28, pp 89-93,2011

[3] Terzo O., Mossucca L., Cucca M., Notarpietro R., Data intensive scientific analysis with Grid Computing,
International Journal of Applied Mathematics and Computer Science, Int. J. Appl. Math. Comput. Sci., Vol. 21, No. 2, 219-228 DOI: 10.2478/v10006-011-0016-z June, 2011

[4] Mossucca L., Terzo O., Cucca M., Notarpietro R., Improvement of job scheduling for automatic chain processing in radio occultation context, The Seventh International Conference on Networking and Services - ICNS2011 Venice , Italy, May 22-27, pp 26-31, 2011

[5] O. Terzo, L. Mossucca, K. Goga , P. Ruiu, and G. Caragnano,, Enhancing Job Scheduling of an Atmospheric Intensive Data Application, International Journal On Advances in Telecommunications, IARIA Online July 2012 vol. 5, no. 1 ,2, pp. 11-20, 2012

[6] POSTER, Perona G., Notarpietro R., Molinaro M., Casotto S., Zoccarato P., Nardo A., Cucca M., Paolella, S. Bordi I., Sol P., Sutera A., Tartaglione N., Speranza A., Nava B., Radicella S., Terzo O., Mossucca L., Gallipoli A., Tataranni F., De Cosmo V., Catalano V., Radio Occultation with ROSA on-board OCEANSAT-2: MISSION overview and lrst results,
International Workshop on Occultation for Probing Atmosphere and Climate, OPAC 2010 Graz, Austria, September 6-11,

[7] POSTER, Mossucca L., Terzo O., Cucca M., Notarpietro R.,, Radio Occultation with ROSA on-board OCEANSAT-2 Grid Environment, International Workshop

on Occultation for Probing Atmosphere and Climate OPAC 2010 Graz, Austria September 6-11, 2010

[8] Terzo O., Mossucca L., Cucca M., Notarpietro R.,, Grid Computing environment to characterize atmospheric profiles, The 4th International Conference on Complex, Intelligent and Software Intensive Systems - CISIS-2010 - Fourth International Workshop on P2P, Parallel, Grid and Internet Computing (3PGIC-2010) Cracow, Poland, February 15-18, pp451-456 2010

[9] Terzo O., Mossucca L., Perona G., Molinaro M., Notarpietro R., Cucca M., Paris D., Lazzara F., The Italian ROSA receiver: prototype Ground Segment for the data processing through Grid Computing techniques, Rivista Italiana di Telerilevamento, Italian Journal of Remote Sensing, 41(3), pag. 109-122 September, 2009, Volume: 41; doi: 10.5721/ItJRS20094139; http://www.aitjournal.com/articleView.aspx?ID=131

[10] Mossucca L., Terzo O., Cucca M., Notarpietro R.,, Atmospheric Remote Sensing Data Processing Through Grid Computing Technique, The 2009 International Conference on Grid Computing and Applications - GCA'09 Las Vegas, Nevada, July 13-16, pp 66-72, 2009

[11] Terzo O., Mossucca L., Ruiu P., Molinaro M., Notarpietro R., Cucca M., Stigliano S.,, A distributed environment approach for the GPS Radio Occultation analysis, InfoSys 2009 Workshop The Fifth International Conference on Networking and Services -ICNS 2009 Valencia, Spain, April 20-25, pp.487-492, 2009

[12] Mossucca L, Terzo O, Molinaro M, Perona G, Cucca M, Notarpietro R,, Results for atmospheric remote sensing data processing through Grid Computing, (HPCS 2010), Caen France June 28 - July 2010, Vol. Proceeding, DOI: 10.1109/HPCS.2010.5547059

[13] O. Terzo, L. Mossucca, K. Goga , P. Ruiu, and G. Caragnano,, Job Scheduler Improvement for Data Intensive Computing, International Journal On Advances in Telecommunications, IARIA

[14] TRANSACTION (at press time), O. Terzo, L. Mossucca, P. Ruiu, G. Caragnano, K. Goga, R. Notarpietro and M. Cucca, Improving Scalability of a Hybrid Infrastructure for E-Science Applications,
Transactions on CCI X, LNCS 7776, pp. 120134 Springer (online).

[15] M. A. Francavilla, M. Righero, F. Vipiana, O. Terzo, P. Ruiu, L. Mossucca, G. Vecchi,, Issues in Compressive Domain Decomposition for Integral Equations, The 6th European Conference on Antennas and Propagation, Prague, Czech Republic, 26-30 March, 2012

[16] O. Terzo, P. Ruiu, L. Mossucca, M. A. Francavilla, F. Vipiana,, Grid Infrastructure for Domain Decomposition Methods in Computational ElectroMagnetics, Chapter: Intech, Grid Computing 2012

[17] O. Terzo, P. Ruiu, L. Mossucca, M. A. Francavilla, F. Vipiana,, Low-cost High Performance Grid Infrastructure for Domain-Decomposition Methods in Computational Electromagnetics, Sixth international on P2P, Parallel, Grid, Cloud and Internet Computing Barcellona, Spain, October 26-28, 2011

[18] Matteo Alessandro Francavilla (Istituto Superiore Mario Boella, Italy), Francesca Vipiana (Politecnico di Torino, Italy), Olivier Terzo, Pietro Ruiu, Lorenzo Mossucca, Giuseppe Caragnano (Istituto Superiore Mario Boella), Giuseppe Vecchi Politecnico di Torino, Italy) ,Scalable Cloud Computing Infrastructure for Electromagnetic Virtual Prototyping 7th EUROPEAN CONFERENCE ON ANTENNAS AND PROPAGATION Gothenburg Sweden 8-12 April 2013

[19] L. Mossucca, O. Terzo, K. Goga, A. Acquaviva, F. Abate, R. Provenzano,, NGS workflow Optimization using a Hybrid Cloud Infrastructure, International Journal On Advances in Networks and Services, IARIA Online December 2012 vol. 5 n 3 ,4

[20] O. Terzo, L. Mossucca, A. Acquaviva, F. Abate, E. Ficarra, R. Provenzano, Optimizing Splicing Junction Detection in Next Generation Sequencing Data on a Virtual-GRID Infrastructure, The 6th International Conference on Bioinformatics and Biomedical Engineering Shanghai, China May 17-20, 2012

[21] O. Terzo, L. Mossucca, A. Acquaviva, F. Abate, R. Provenzano,, A Cloud Infrastructure for Optimization of a Massive Parallel Sequencing Workflow, The 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Ottawa, Canada May 13-16, 2012 pp. 678-679

[22] O. Terzo, L. Mossucca, A. Acquaviva, F. Abate, R. Provenzano,, Virtual Environment for Next Generation Sequencing Analysis, The Eighth International Conference on Networking and Services, St. Maarten, Netherlands Antilles March 25-30, 2012 pp. 47-51

[23] O. Terzo, L. Mossucca, A. Acquaviva, F. Abate, E. Ficarra, R. Provenzano R,, Reverse Engineering of TopHat: Splice Junction Mapper for Improving Computational Aspect, Third International Conference on Intelligent Systems, Modelling and Simulation Kota Kinabalu, Malaysia February 8-10, 2012 pp. 171-176

[24] POSTER, G. Donvito, D. Cesini, L. Gaido L. Mossucca, O. Terzo, P. Ruiu and A. Acquaviva,, TopHat on the Grid: an automatic workflow for sequence alignment exploiting EGI/IGI grid infrastructure, EGI Technical Forum 2012, Prague, Czech Republic September 17-21, 2012

[25] Terzo O., Mossucca L., Ruiu P., Abate F., Acquaviva A., Ficarra E., Macii E., Next Generation Sequencing on Hybrid Grid Cloud Infrastructure, World Congress on Engineering and Technology (CET2011) Shanghai, China October 28-30, 2011 pp. 572-575

[26] Terzo O., Mossucca L., Ruiu P., Abate F., Acquaviva A., Ficarra E., Macii E.,, An effective grid infrastructure for efficiently support high throughput sequencing analysis, The 2nd International Conference on Engineering and Meta-Engineering: ICEME 2011, Orlando, Florida (USA) March 27-30, 2011 pp. 40-44